NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

SIMULATION OF FLIGHT OPERATIONS AND PILOT DUTIES IN LANTIRN FIGHTER SQUADRONS USING SIMKIT

by

Mustafa Azimetli

June 2008

Thesis Advisor:	Arnold Buss
Second Reader:	Susan M. Sanchez

Approved for public release; distribution is unlimited

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.			
1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** June 2008	**3. REPORT TYPE AND DATES COVERED** Master's Thesis	
4. TITLE AND SUBTITLE Simulation of Flight Operations and Pilot Duties in LANTIRN Squadrons Using Simkit		**5. FUNDING NUMBERS**	
6. AUTHOR(S) Mustafa Azimetli			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000		**8. PERFORMING ORGANIZATION REPORT NUMBER**	
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A		**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**	
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.			
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.		**12b. DISTRIBUTION CODE**	

13. ABSTRACT (maximum 200 words)

The LANTIRN (Low-Altitude Navigation and Targeting Infra-Red for Night) introduces important around-the-clock strike capability to air forces. At the same time, it strains pilot manpower requirements. The Turkish Air Force asked for a simulation tool that would find the necessary number of pilots and their qualifications for LANTIRN squadrons under different operations scenarios. This thesis develops a simulation that satisfies this request. The simulation takes the pilot ground duties as well as flight operations into account. The weather model inside the simulation introduces the effects of weather conditions around airfield. The model was implemented in the Java language, using the Simkit library for the discrete event simulation. The user interacts with a Graphical User Interface (GUI) to define the parameters, experiment input factors, and sizes. The output is a data table with required pilot number and qualifications. Because access to certain classified data is not possible, the thesis sets a general guideline for future analyses that would have the actual data. This study uses notional but realistic values for the parameters and input factor levels. Using the resulting output table, future analysts can expand and tailor the levels of analyses.

14. SUBJECT TERMS LANTIRN, Simulation, Pilot Manpower, Simkit, GUI, Design of Experiment, NOLH, Regression Analysis			**15. NUMBER OF PAGES** 107
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UU

NSN 7540-01-280-5500 Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39.18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

SIMULATION OF FLIGHT OPERATIONS AND PILOT DUTIES IN LANTIRN FIGHTER SQUADRONS USING SIMKIT

Mustafa Azimetli
Lieutenant, Turkish Air Force
B.S., Turkish Air Force Academy, 1999

Submitted in partial fulfillment of the
requirements for the degree of

**MASTER OF SCIENCE IN MODELING,
VIRTUAL ENVIRONMENTS, AND SIMULATION**

from the

**NAVAL POSTGRADUATE SCHOOL
June 2008**

Author: Mustafa Azimetli

Approved by: Arnold Buss
Thesis Advisor

Susan M. Sanchez
Second Reader

Mathias Kolsch
Chair, MOVES Academic Committee

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

The LANTIRN (Low-Altitude Navigation and Targeting Infra-Red for Night) introduces important around-the-clock strike capability to air forces. At the same time, it strains pilot manpower requirements. The Turkish Air Force asked for a simulation tool that would find the necessary number of pilots and their qualifications for LANTIRN squadrons under different operations scenarios. This thesis develops a simulation that satisfies this request. The simulation takes the pilot ground duties as well as flight operations into account. The weather model inside the simulation introduces the effects of weather conditions around airfield. The model was implemented in the Java language, using the Simkit library for the discrete event simulation. The user interacts with a Graphical Use Interface (GUI) to define the parameters, experiment input factors, and sizes. The output is a data table with required pilot number and qualifications. Because access to certain classified data is not possible, the thesis sets a general guideline for future analyses that would have the actual data. This study uses notional but realistic values for the parameters and input factor levels. Using the resulting output table, future analysts can expand and tailor the levels of analyses.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ABBREVIATIONS AND ACRONYMS

LANTIRN	Low-Altitude Navigation and Targeting Infra-Red for Night
MANTIRN	Medium-Altitude Navigation and Targeting Infra-Red for Night
PGM	Precision Guided Munitions
TFR	Terrain-Following Radar
FLIR	Forward-Looking Infra-Red
HUD	Head Up Display
MFD	Multi-Function Display
MSA	Minimum Safe Altitude
IFR	Instrument Flight Rules
AGL	Above Ground Level
GUI	Graphical User Interface
DES	Discrete Event Simulation
FEL	Future Event List
SOF	Supervisor of Flight
RSU	Runway Supervisory Unit
BFO	Base Flight Operations
BWOC	Base War Operations Center
MCR	MANTIRN Combat Readiness
LCR	LANTIRN Combat Readiness
METAR	Aviation Routine Weather Report
ICAO	International Civil Aviation Organization
AR	Auto Regressive

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

First, I want to thank my country, Turkey, and the Turkish Air Force for providing me with this excellent opportunity of being a student at the Naval Postgraduate School. It was an outstanding experience being here.

I would like to thank Professor Arnold Buss for his guidance and help during the preparation of this thesis. I also would like to thank Professors Enver Yucesan and Susan Sanchez for their valuable instruction and help. Their guidance and valuable inputs helped me stay focused.

Finally, I would like to express my appreciation and gratitude to my wife, Sevda, for her patience, understanding, and support throughout my study. Her presence made it possible to finish this thesis. She is the cornerstone of my family.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

A. BACKGROUND

1. LANTIRN System

The LANTIRN stands for Low-Altitude Navigation and Targeting Infra-Red for Night. The LANTIRN system consists of a navigation pod (AN/AAQ-13) and a targeting pod (AN/AAQ-14) (Jane's Avionics, 2006). The F-15E, F-16, and F-14 have the capability to use this system. The LANTIRN enables a host aircraft to penetrate enemy airspace at extremely low altitudes at high speeds under reduced visibility or bad weather conditions, employ precision guided munitions (PGM) like laser-guided freefall bombs or AGM-65 MAVERICK missiles, and return to base safely. Figure 1 shows an F-16 aircraft carrying both the navigation and targeting pods.

Figure 1. F-16 Aircraft Carrying the LANTIRN System [From (Jane's Avionics, 2006)].

The AN/AAQ-13 navigation pod contains a Forward-Looking Infra-red (FLIR) unit and a Terrain- Following Radar (TFR). The TFR interacts with the flight control system of the aircraft, and enables a fully automatic, very low-altitude navigation capability at night or during bad weather. The FLIR unit provides night vision for the pilot, and this FLIR imagery can be viewed on Heads-Up Displays (HUD) and Multi-Function Displays (MFD).

The AN/AAQ-14 targeting pod contains a targeting FLIR and a laser designator. The targeting pod captures video imagery of a target area and pilots view this imagery on MFD. Pilots designate targets of interest. If a pilot is employing a laser-guided weapon, then the targeting pod designates the target with a laser beam, enabling a bomb to follow the laser beam to the target. If the pilot is employing an AGM-65 MAVERICK missile, the targeting pod hands the target off to the missile. The AGM-65 is a fire-and-forget, stand-off weapon.

2. MANTIRN System

The MANTIRN stands for Medium-Altitude Navigation and Targeting Infra-Red for Night. The host aircraft does not have to carry both pods. It can carry either pod and utilize capabilities of that pod. Use of all the capabilities of the LANTIRN system requires a high degree of pilot proficiency. The Turkish Air Force requires the pilot to spend at least two training years in a squadron before starting LANTIRN training. This equates to almost 500 flight hours. To enable pilots to use limited capabilities of the LANTIRN system before they are cleared to fly the complete system, the Turkish Air Force uses the MANTIRN system.

In the MANTIRN configuration, a pilot uses only the targeting pod. The navigation pod may be on board; however, a pilot does not use the capabilities of that pod. He can manually fly at low altitude during day conditions. A pilot can only fly above the Minimum Safe Altitude (MSA) during night conditions. A pilot uses the full capability of the targeting pod under allowed conditions. The MANTIRN system enables the squadrons give some LANTIRN training to their pilots. Pilots who do not meet the eligibility criteria for the LANTIRN system can

still employ PGMs under favorable conditions. This, in turn, increases the strike capability of the squadron. MANTIRN training is a step before pilots are entitled to LANTIRN training. Pilots who spend one training year in the squadron can start receiving MANTIRN training.

3. Pilot Qualifications

a. Flight Positions

Fighter aircrafts fly as formations except in a few situations. There are usually either two or four aircraft in a formation. Formations having three aircraft are possible but uncommon. Pilots fly in the formation based on their assigned flight positions. There are three flight positions:

- Wingman
- Two-ship Lead
- Four-ship Lead

Pilots start their flight career as wingmen, which require the lowest level of experience. Wingmen can only fly as number two or number four in the formations unless they are in upgrade training to flight lead status. As stated in the AFI 11-2F-16, wingmen “help the leader plan and organize the mission. They have visual lookout and radar responsibilities, perform back-up navigation tasks, and are essential to target destruction objectives. Wingmen engage as briefed or when directed by the leader and support when the leader engages.” (AFI 11-2F-16, 10 May 1996) When wingmen accumulate 500 flight hours as a primary pilot, they can receive upgrade training to two-ship lead status.

Number one and number three pilots have to be flight leads. AFI 11-2F-16 defines flight leads: “Flight leaders have the general responsibility for planning and organizing the mission, leading the flight, delegating tasks within the flight, and ensuring mission accomplishment.” (AFI 11-2F-16, 10 May 1996) There are two types of flight leads: two-ship leads and four-ship leads. Two-ship

leads normally fly in the number-one position in a two-ship formation or the number-three position in a four-ship formation. They can also fly in wingman position. When two-ship lead pilots accumulate 750 flight hours as a primary pilot, they are entitled to receive four-ship lead status. Four-ship lead pilots lead formations of any number of aircraft. They normally fly as number one, but also can fly in any position within the formation. Table 1 shows the pilot upgrade criteria for flight positions and Instrument Flight Rules (IFR) weather categories.

Table 1. Pilot Upgrade Criteria Table.

Pilot Status	IFR Category	Pilot Criteria	
		Primary Pilot (Hours)	Type Flight (Hours)
Wingman	III	100	50
Two-Ship Lead	II	500	100
Four-Ship Lead	I	750	150

b. LANTIRN and MANTIRN Categories

In a similar fashion to flight positions, pilots have categories for MANTIRN and LANTIRN missions. There are two categories for MANTIRN:

- MANTIRN Category 5
- MANTIRN Category 4

There are three categories for LANTIRN:

- LANTIRN Category 3
- LANTIRN Category 2
- LANTIRN Category 1

When an inexperienced pilot joins a LANTIRN squadron as first assignment, no category is designated. Pilots in this status first receive MANTIRN training. In order to start MANTIRN category training, pilots have to spend at least one training year in the squadron. After MANTIRN combat readiness training, pilots become MANTIRN category-five (Cat-5) pilots. Cat-5 pilots can only execute medium altitude level attacks at night. They can fly level and/or FLIR-assisted diving attacks from medium level in daylight conditions. Pilots start receiving Cat-4 training after their combat readiness training is over. MANTIRN Cat-4 training takes around two months. After the check flight, they become MANTIRN category-four (Cat-4) pilots. Cat-4 pilots can fly FLIR-assisted attacks above the MSA and can execute AGM-65 attacks. Pilots stay in this category until they receive LANTIRN training.

Pilots can receive LANTIRN training after spending at least two training years in the squadron. When the pilot finishes LANTIRN combat readiness training, he becomes a LANTIRN category-three (Cat-3) pilot. LANTIRN Cat-3 pilots cannot fly loft attack profiles during day or night. Otherwise, they are unlimited. After pilots complete their Cat-3 training, they start LANTIRN category-two (Cat-2) training. Cat-2 pilots cannot fly loft attack profiles at night. The last training phase is category-1 (Cat-1) training. LANTIRN Cat-1 training takes about four months to complete. At the end of the training, the pilot is entitled as Cat-1 pilot. He is unlimited in terms of LANTIRN operations.

Fighter pilots join their squadron after their training in their aircraft type (this is their first assignment) or after they complete their assignment in another squadron. In the first case, they are inexperienced wingmen with approximately 80 flight hours in F-16s. They have no MANTIRN or LANTIRN category. In the second case, they are four-ship leads with more than 750 flight hours. If the first assignment of these four-ship leads was a LANTIRN squadron, they will join the squadron as LANTIRN category-one (Cat-1) pilots. If their first assignment was not a LANTIRN squadron, they will have no LANTIRN category.

In this case, these experienced pilots do not receive MANTIRN training and start their LANTIRN training after their orientation period is over.

c. IFR Weather Categories

Pilots also have categories for Instrument Flight Rules (IFR) weather conditions. There are three weather categories for the pilots:

- IFR weather category-three (IFR Cat-3)
- IFR weather category-two (IFR Cat-2)
- IFR weather category-one (IFR Cat-1)

When pilots finish their F-16 training and join their squadron, they have IFR Cat-3 category clearance. In order for them to take off and land at an airfield, visibility has to be three kilometers or better and the ceiling has to be 1000 feet above ground level (AGL) or higher. After logging 500 flight hours as primary pilots, pilots upgrade to IFR category-two. Visibility has to be two kilometers or better and the ceiling has to be 500 feet AGL or higher. After 750 flight hours, pilots upgrade to IFR Cat-1. They can fly in any weather condition as long as visibility is above 800 meters and the ceiling is 200 feet AGL or higher. If weather conditions are below respected category minimums, then pilots in this category do not fly. A more in-depth description is in Chapter III.

B. RELATED RESEARCH

German developed a knowledge-based, discrete-event simulation named STAR-Eagle (German, 1990). This is a decision-support tool developed for the Alaskan Air Command. It simulates sortie production at Galena and King Salmon, two forward-operating locations in Alaska for the F-15 aircraft. The STAR-Eagle tool takes the number of aircraft, munitions quantity, fuel quantity, the number of flight crews, the number of maintenance crews, weather conditions, mechanical attrition rate for the aircraft, and hostile attrition rate as simulation input. It then calculates the number of sorties that can be flown for

each day for three different scenarios. These scenarios are peacetime, operations readiness inspection, and crisis conditions. The user can also change input variables and see the effects of these changes on sortie production rate. STAR-Eagle uses an event list to run the simulation.

Eagle View is a simulation tool designed to help the wing-level commanders with their decision making process (Zahn & Renken, 1997). This tool gives an “eagle-eye” view of the current operations on the base. (The author states this attribute of the simulation is inactive due to technical constraints. Instead, another simulation tool called data injector produces events and injects those events into a real-time simulation as if they are coming from real world.) When the wing commander wishes to implement a new plan for the base operations instead of the current plan, he can simulate this new plan, see its effects, and make a decision accordingly. For example, the wing commander receives an order for deployment of ten aircraft for two weeks: Can the wing get ready in 24 hours for this deployment? He can retrieve the answer by running this new plan in the simulation tool. The program takes the current real-time conditions as the starting conditions and runs simulations to produce the answer. This simulation tool uses object-oriented discrete event simulation.

Brown and Powers developed a generic simulation tool that would produce the resources necessary to support flying schedules (Brown & Powers, 2000). Its intended user population is the maintenance community. The simulation takes the flight schedule as input. It then runs a stochastic simulation to calculate necessary maintenance resources to accomplish this flight schedule. Aircraft parts break and require repair. Aircraft also go into scheduled maintenance. Certain types of missions affect the number of crew chiefs needed. The simulation ends after running for a certain number of days. As output, it gives required resources and statistics for resource use.

Harris developed a simulation tool that can produce sortie generation rates (Harris, 2002). Sortie Generation Rate (SGR) is a generic simulation tool that takes inputs from the user via a graphical user interface (GUI). It runs the

simulation using the Arena simulation tool. It is a rich model with detailed inputs and detailed modeling of flight operations. Some constraints that Harris models involve the number of aircraft, number of aircrew, number of maintenance crew, crew rest periods, ground and air abort rates, aircrews on leave or sick, durations for sortie related events, etc. The simulation uses the discrete event simulation technique to produce the number of sorties that a wing can fly over in a certain period.

C. PROBLEM STATEMENT AND GOAL OF THESIS

Introduction of the LANTIRN system enabled air forces to operate at night as well as during the day. The air forces increased their effectiveness because of the use of surprise and night masking of operations. Night operations saw a great increase in recent operations in the Gulf War and Kosovo. On the other hand, this increase in night flights strained pilot resources. In previous operations, pilots flew during the day and rested at night. They were ready the next morning for the operations. Today, air forces operate around the clock. They need more pilots for continuous 24-hour operations. There should be enough pilots to make day and night shifts. There is a need for the pilot manpower planning for continuous operations.

Turkish Air Force Headquarters asked for a planning tool that would find the required number of pilots in a LANTIRN squadron for various operations scenarios using simulation techniques. This simulation tool would produce the number and composition of pilots under certain constraints. The meaning of “composition” here is the number and the ratios of pilots based on flight positions, LANTIRN and MANTIRN categories, and IFR weather categories. Aforementioned simulation tools took the number of aircrews as an input and tried to find the required maintenance resources or tried to find the number of sorties that could be flown. By problem definition, this thesis takes the sortie rates as one of the inputs and tries to find the required number of aircrew for

operations. It models duties of pilots other than flight operations. These duties, which relate to flight operations, consume pilot resources.

This thesis used the discrete event simulation technique. The simulation utilized Java and the Simkit simulation tool developed by Professor Buss. The simulation experiment is used to produce simulation results. The simulation tool has a GUI that allows a user to manipulate fixed parameters and experiment factor inputs. The reason is that by the nature of the research problem, most of the input data is inaccessible or unknown to researchers. To overcome this difficulty, a user interacts with GUI for inputs, runs the simulation, and gets the results in a comma-separated file for further analysis. The design of the analysis chapter is in such a way that it constitutes an example to future users for the analysis that they can make.

THIS PAGE INTENTIONALLY LEFT BLANK

II. MODEL

A. DISCRETE-EVENT SIMULATION, EVENT GRAPHS, AND SIMKIT

Discrete-Event Simulation (DES) is a simulation methodology where state variables change instantaneously at separate points in time (Law, 2007). There are four basic components of DES: event, Future Event List (FEL), state variables, and parameters (Buss, Summer 2007).

- Event: Events occur at discrete points in time. Law defines the event as an "instantaneous occurrence that may change the state of the system" (Law, 2007). The events do not have to inflict a change in the system. Every event has an associated event time. When that time arrives, the event is executed. An event may schedule additional events.
- Future Event List (FEL): FEL is a list of scheduled events sorted in time order. It is a "to-do" list of scheduled events (Buss, 1995). FEL changes dynamically during the execution of a simulation. FEL contains events and their scheduled times. When FEL gets empty (i.e., all scheduled events are executed), DES ends.
- State variables: These variables define the state of the system at a particular point in time. Systems are collection of entities and entities have attributes. These attributes are the part of the system state (Law, 2007). State variables change values at a countable number of times (Buss, Summer 2007).
- Parameters: Parameters are the constants that do not change in the course of the simulation. They remain constant, unlike the state variables. Streams of random variables are also treated as constant parameters.

The event list and manipulation of the event list runs the Discrete-Event Simulation. The following graph is taken from the OA3302 System Simulation class notes and it shows the basic DES algorithm:

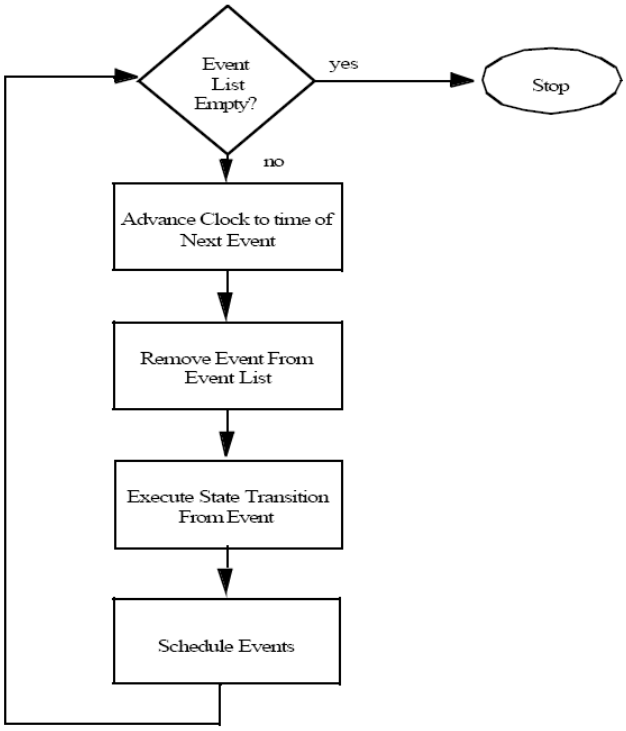


Figure 2. Basic Discrete Event Algorithm [From (Buss, Summer 2007)]

Event graphs "is a way of representing the FEL logic for a discrete event model." (Buss, 1995) Two major components of the event graphs are nodes and edges. Nodes represent the events. Edges represent scheduling of other events. Each node does not have to have an edge that schedules an event. Edges optionally might have a time delay and/or Boolean condition. Figure 3 shows a basic event graph with argument. We interpret the graph as follows: "The occurrence of event A causes event B to be scheduled after a time delay of *t*, providing condition (i) is true (after the state transitions for event A have been made)" (Buss, 1995).

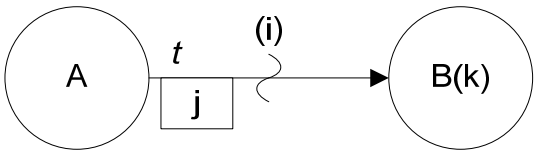


Figure 3. Basic Event Graph With Argument [From (Buss, Summer 2007)]

By convention, time delay *t* is placed by the tail of scheduling edge. *t* may be omitted in the case of no time delay (i.e., *t* equals zero). The edge condition is above the wavy line. If the edge condition is true, then the scheduling of event B takes place. The passing parameters are an important feature of event graphs. “This enables information about the simulation’s state at a particular simulation time to be transmitted to a future event in a kind of ‘time capsule’” (Buss, 2001). Parameters are placed in a box on event graphs. The receiving event must have a corresponding argument that matches the parameter. In Figure 3 above, event A passes parameter j to the receiving event B. Event B receives parameter j by setting the value of argument k to the value passed by parameter j.

This model implements DES by using Simkit. Simkit is a software package written by Professor Arnold Buss in the Java language (Buss, 2001). The modeler interacts with FEL by using classes that inherit abstract class SimEntityBase in Simkit. Each node corresponds to a method beginning with the string “do.” Scheduling edges are implemented by the call of waitDelay() method. If there is a Boolean edge, waitDelay() call is wrapped inside an “if” test. The waitDelay() method call takes some arguments depending on the condition. The simplest signature of the waitDelay() method is (String, double). String represents the name of the event, and double is the amount of time delay. If the scheduling event is passing an object, then it is added as the third parameter to the signature. The following code snippet is the corresponding Simkit implementation of Figure 3:

```
public void doA() {
        int j = . . .;
        // State transitions for Event A
        if (i) {
                waitDelay("B", t, j);
        }
}
public void doB(int k) {
        // State transitions for Event B.
}
```

(Buss, Summer 2007). This thesis also used component-based simulation modeling. Simulation components communicate using the Listener Pattern (Buss, 2000). In a listener pattern, there are two types of components: a listener component and an event source component. Many listeners might listen for a single source component. At the same time, a listener might listen for many sources. When an event is fired in the source component, the registered listeners hear that and schedule related events if they have a "do" method with the same signature. "A Listener processes hears an event as if it had scheduled it." (Buss, Fall 2007). The following graph shows a SimEvent listener pattern:

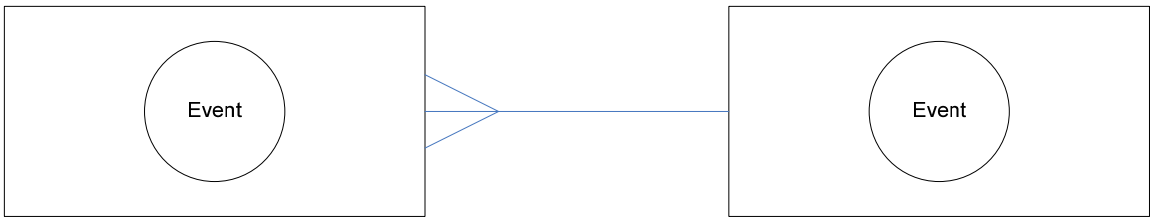


Figure 4. Listener Pattern [From (Buss, Fall 2007)].

The stethoscope-shaped connector between the components indicates that component B is listening to component A for the event named “Event” to happen. The reader should note that event names and signatures have to be the same in both components for the listener pattern to work. If we desire an event name change in the listener component, an adapter pattern can be used. Adapter patterns use the same principles with listener patterns, and they can be modeled with listener patterns. In adapter patterns, when Event A in source occurs, Event B in listener is triggered. The following graph shows an adapter pattern:

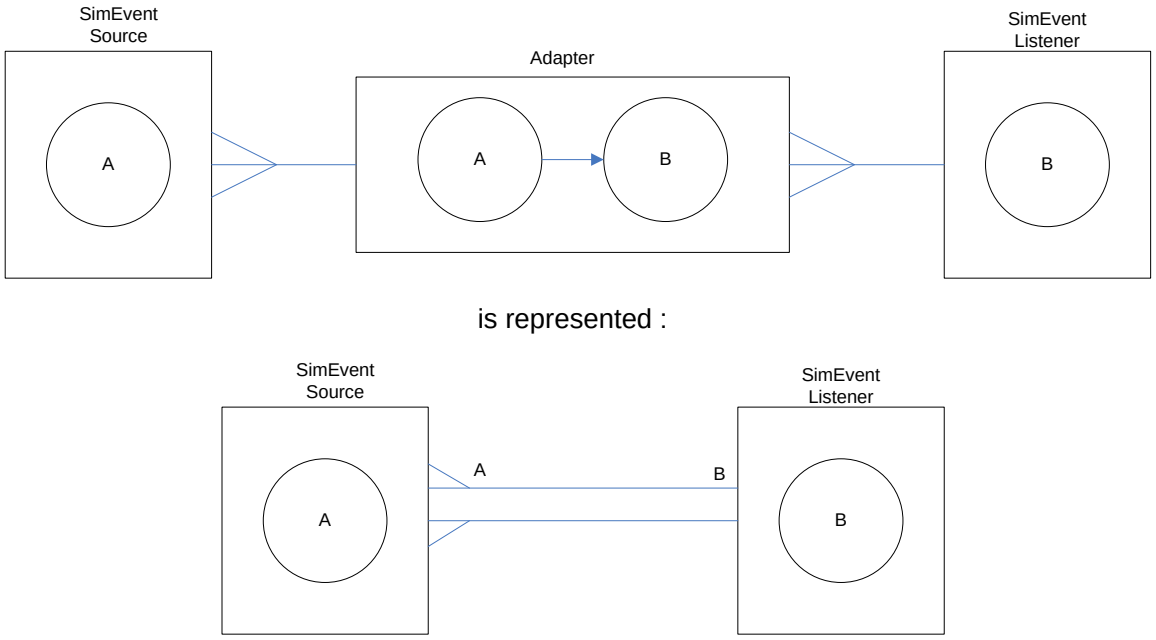


Figure 5. Adapter Pattern and Representation [From (Buss, Fall 2007)].

The model in this thesis implements both the listener and the adapter patterns.

B. SIMULATION COMPONENTS

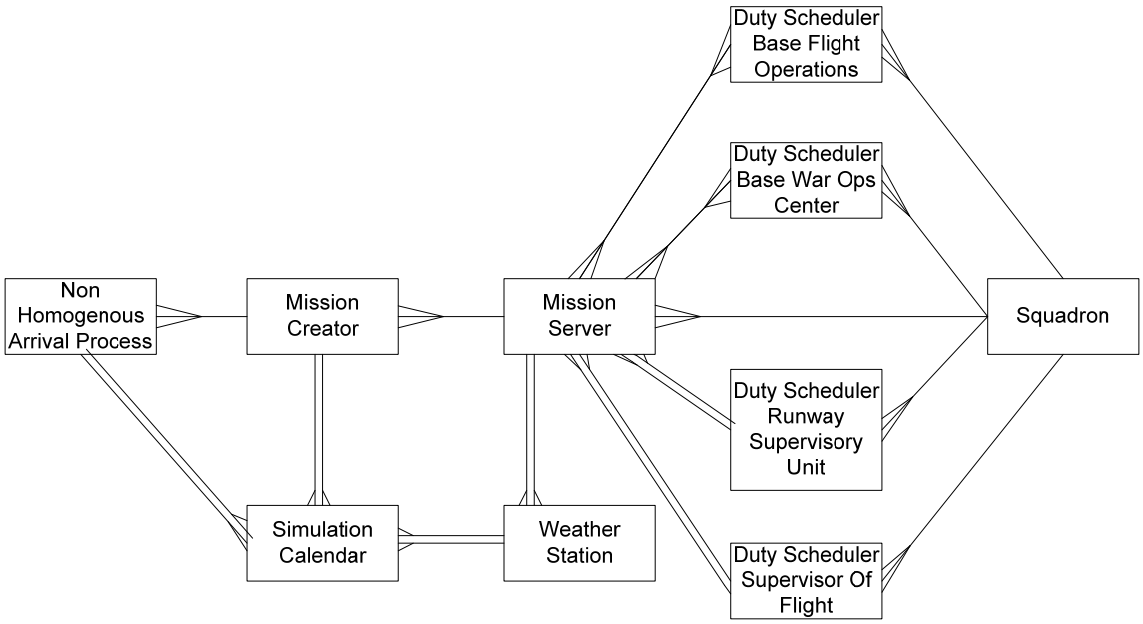


Figure 6. Simulation Components with Listener and Adapter Patterns.

Figure 6 shows the main components of the model used in the simulation with listener and adapter patterns. The simulation model operates like a simple server model. In a simple server model, an arrival process produces arrival events. When the job creator hears this arrival event, it produces a job and passes it to server. The server then takes the job and accomplishes the service. The model in the above figure is more complex than the simple server model but utilizes the same idea. We first introduce the general model and then describe the detailed flow of operations within each module.

In real life, there is a planner in headquarters. This planner develops the overall operations plans. He plans the operations by using many aircraft from different squadrons. Flights from different squadrons are put together to form packages. Each package has its own overall mission objective. Flights within the package have their specific positions, roles, and objectives. There might be many packages flying every day during the operations. The Air Tasking Order (ATO) broadcasts this overall flight plan to all squadrons. The scheduler in the squadron

receives this ATO, defines the details of missions (e.g., type of mission, weapons load, etc.), and assigns pilots to flights. On a base, there are duties related to flight operations. Pilots from the squadron carry out these duties in addition to flight operations. The scheduler also assigns pilots to these duties. Pilots who are scheduled to fly a mission prepare a detailed mission profile consistent with the overall package requirements. Pilots brief the missions, take off, land, and return to squadron. After a mission debrief, they are ready for another mission. Every pilot stays active in the squadron for a certain amount of time. That time is usually 12 hours. At the end of 12 hours of working, pilots rest for 12 hours. Some pilots go to duty stations. At the end of their duty cycle, they come back to the squadron building. These duties are explained in detail in the following sections.

All modules in the model are independent from each other. However, they are linked by using listeners and adapters. They can run individually, but they need to interact to produce the simulation results. When a user starts the simulation, the SimulationCalendar starts providing real-world calendar data for other components. The simulation starts at either sunrise or sunset time on a random day in the year 2010. From then on, with one-hour intervals, the SimulationCalendar class announces real-world time. The NonHomogenousArrivalProcess, MissionCreator, and WeatherStation classes receive this real-world time information, because they are connected to the SimulationCalendar class with connectors. The NonHomogenousArrivalProcess class produces arrivals based on the time of day. The simulation uses different arrival rates (λ) for daytime and nighttime. This class takes the role of planner and ATO process. The MissionCreator class listens to the NonHomogenousArrivalProcess class for arrivals. When an arrival event happens, it triggers the MissionCreator class to produce a mission. MissionCreator determines the type and attributes of mission based on time of day and some stochastic procedures. MissionCreator receives time information from the SimulationCalendar class. The MissionServer class listens to the

MissionCreator class for missions. MissionCreator hands off the mission to the MissionServer class. The MissionServer class does the bulk of the process. This class assigns pilots to missions and makes them fly missions, return to squadron, and debrief. The MissionServer class receives current weather information from the WeatherStation class at the beginning of every hour. It uses this weather information for the scheduling and selection of pilots for missions.

There are four types of flight-related duties. One four-ship lead pilot has to be in the control tower during flight operations. He is called the Supervisor of Flight (SOF). The DutySchedulerSOF class schedules pilots for this duty based on flight schedule. Another pilot has to be in the Runway Supervisory Unit (RSU) located near the approach end of runway. The DutySchedulerRSU class schedules pilots for this duty. Another pilot carries out some tasks for the safe conduct of flight operations on base. The DutySchedulerBFO class schedules pilots for this mission. One pilot from the squadron acts as a representative in the Base War Operations Center (BWOC). He is responsible for coordination with other units on base. DutySchedulerBWOC handles the scheduling of pilots for this duty post. All these duty scheduler classes listen to or are connected to the MissionServer class, because they schedule pilots based on the flight schedule. The Squadron class is the central piece for the management of pilot resources. It listens to MissionServer and all duty scheduler classes. It lends pilots to these classes, gets them back when they complete their mission or duty, and manages their crew rest cycles.

The following sections will discuss further details of each component with their event graphs.

1. Simulation Calendar

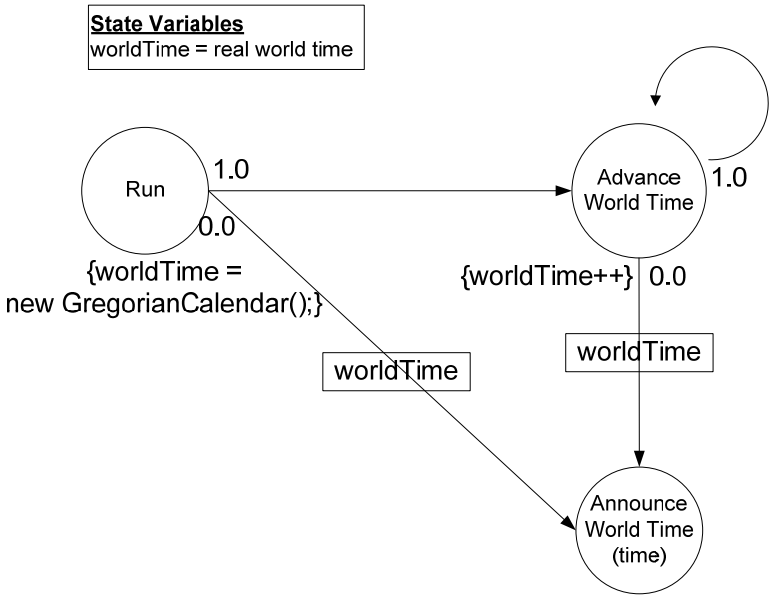


Figure 7. Simulation Calendar Event Graph.

Figure 7 shows the event graph of the SimulationCalendar class. The SimulationCalendar class is the first class to run when the simulation begins. Some components in the simulation need real-world time data together with the simulation time data. The SimulationCalendar class set the real-world time to sunrise or sunset on an arbitrary day in the year 2010. From then on, it advances the clock at one-hour intervals. Every time it advances the clock, it announces the current time. Modules that are tuned to listen for this class hear this announcement and receive the current time. The NonHomogenousArrivalProcess, MissionCreator, and WeatherStation classes are notified when SimulationCalendar class invokes the doAnnounceWorldTime() method.

2. Non Homogenous Arrival Process

A non-homogenous arrival process was used to model the sortie generation in the simulation. The reason is that the LANTIRN squadrons fly

intensively at nights. They still fly during daytime, but the number of sorties flown during daytime is less than flown during nighttime. In a non-homogenous Poisson process, the arrival rate λ is not fixed, but changes as a function of time. In the simulation model, there are two arrival rates: one for nighttime and another for daytime. The simulation divides the number of sorties planned for night by the time duration between sunset and sunrise and calculates night arrival rate. It calculates the day arrival rate in the same manner. The same arrival rate is used for the whole night or day, i.e., arrival rates do not change from one hour to next. Assume the daytime length is 10 hours and the nighttime length is 14 hours for an arbitrary day. Further, assume that four missions are planned for daytime and eight missions are planned for nighttime. In this case:

$$\lambda_{NIGHT} = \frac{8}{14} = 0.57$$

$$\lambda_{DAY} = \frac{4}{10} = 0.4$$

$$\lambda^* = 0.57$$

λ^* is equal to the highest of arrival rates. I use a thinning algorithm to generate arrival times. Figure 8 provides a brief overview of the thinning algorithm. Please refer to Law, page 473, for additional details algorithm (Law, 2007).

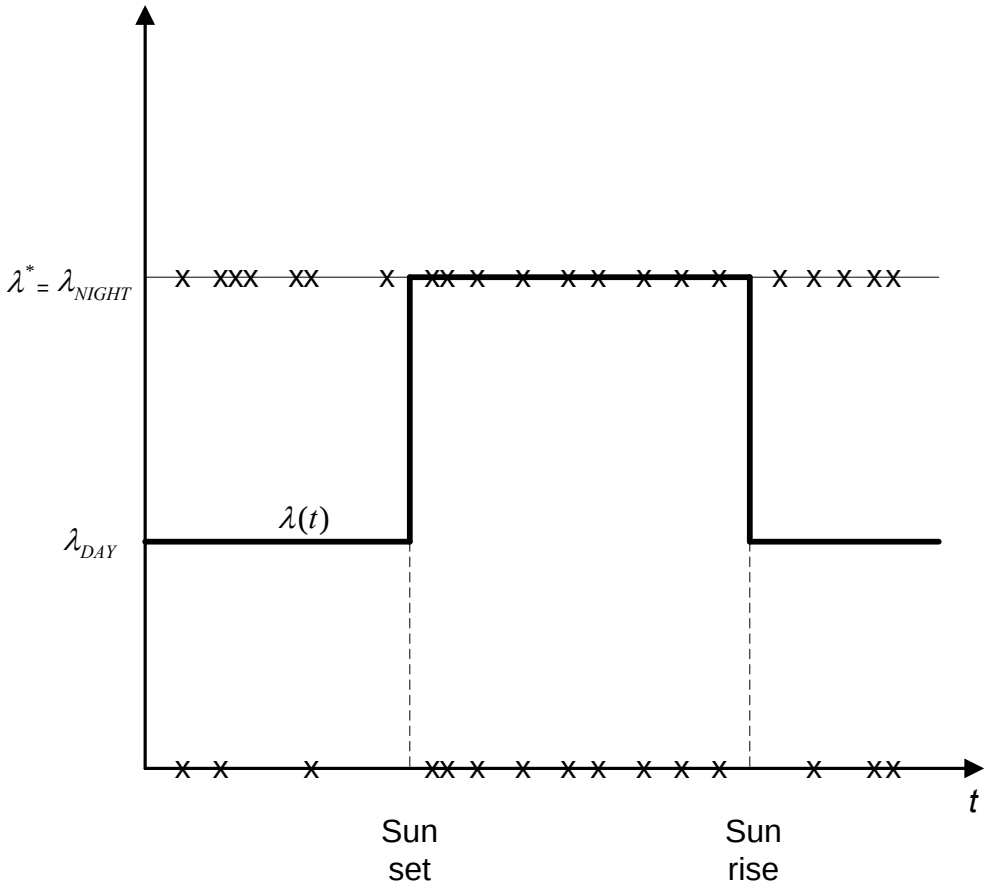


Figure 8. Generating Non-Homogenous Arrivals by a Thinning Algorithm.

Figure 8. shows the generation of non-homogenous arrivals by the use of a thinning algorithm. The algorithm generates arrivals at the highest rate. In the simulation model, there are two rates: λ_{DAY} and λ_{NIGHT}. In the above case, λ_{NIGHT} is bigger than λ_{DAY} rate, so arrivals are generated at the $\lambda^* = \lambda_{NIGHT}$. At each arrival generation, a random number between zero and one is drawn. If that random number is less than the ratio of $\lambda(t)/\lambda^*$, then this arrival is accepted; otherwise, it is rejected. In the figure above, a reader can observe that some of the day arrivals will be rejected, and all of the night arrivals will be accepted. This algorithm enables the simulation to change the arrival rates between day and night conditions.

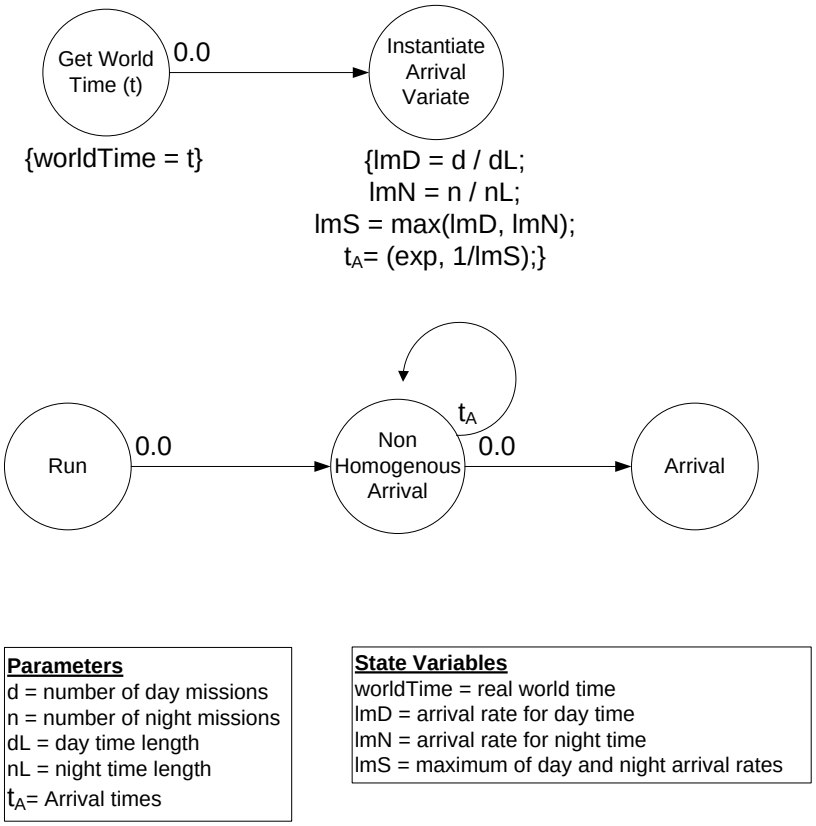


Figure 9. Non-homogenous Arrival Event Graph.

Figure 9. shows the event graph for the NonHomogenousArrivalProcess class. An adapter connects this class to the SimulationCalendar class. When the SimulationCalendar class invokes its doAnnounceWorldTime() method, the doGetWorldTime() method in NonHomogenousArrivalProcess is also invoked. This method then invokes the doInstantiateArrivalVariate() method. This method calculates the lengths of day and night, λ^*, λ_{DAY}, and λ_{NIGHT} values. Based on these values, it instantiates an arrival process variable that will be used to produce arrivals in the doNonHomogenousArrival() method. The doNonHomogenousArrival() method produces non-homogenous arrivals using the thinning algorithm as described above. It then invokes the doArrival() method every time it produces an arrival event. This step is for the compatibility with the MissionCreator class. The MissionCreator class listens for a doArrival() method invocation.

3. Mission Creator

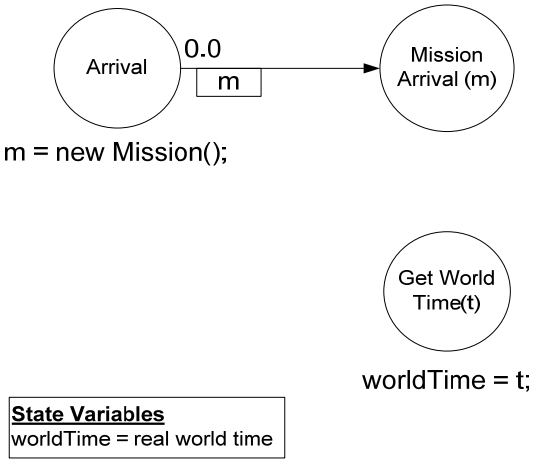


Figure 10. Mission Creator Event Graph.

Figure 10 depicts the event graph of the MissionCreator class. The MissionCreator class listens to the SimulationCalendar and NonHomogenousArrivalProcess classes. The doGetWorldTime() method is invoked when MissionCreator hears the AnnounceWorldTime event fired by the SimulationCalendar class. This method continuously updates the current world time. The MissionCreator class uses the current world time during the creation of new mission objects. MissionCreator's doArrival() method is invoked when NonHomogenousArrivalProcess fires an Arrival event. In this doArrival() method, a new mission gets created. Stochastic processes determine attributes of a new mission object. There are parameters that the user enters using the GUI of the program. These parameters are sortie duration, return to squadron duration, and debriefing duration. The user enters the minimum, maximum, and mode values. The model uses these values to form triangular distributions for related event durations. For every mission object that is created, the simulation produces random values for the events using triangular distribution parameters and writes them to the object. The simulation also determines the type of mission using stochastic procedures. The type of mission can be LANTIRN, MANTIRN, or SAT_DAY (Surface Attack Tactics, Day). Time of day and user-entered factors

also contribute to this process. The simulation then writes all these values into related fields of the object. The doArrival() method invokes the doMissionArrival() method and passes the mission object. The doMissionArrival() method is in the MissionServer class. The MissionServer class listens to the MissionCreator class and receives the mission object.

4. Mission Server

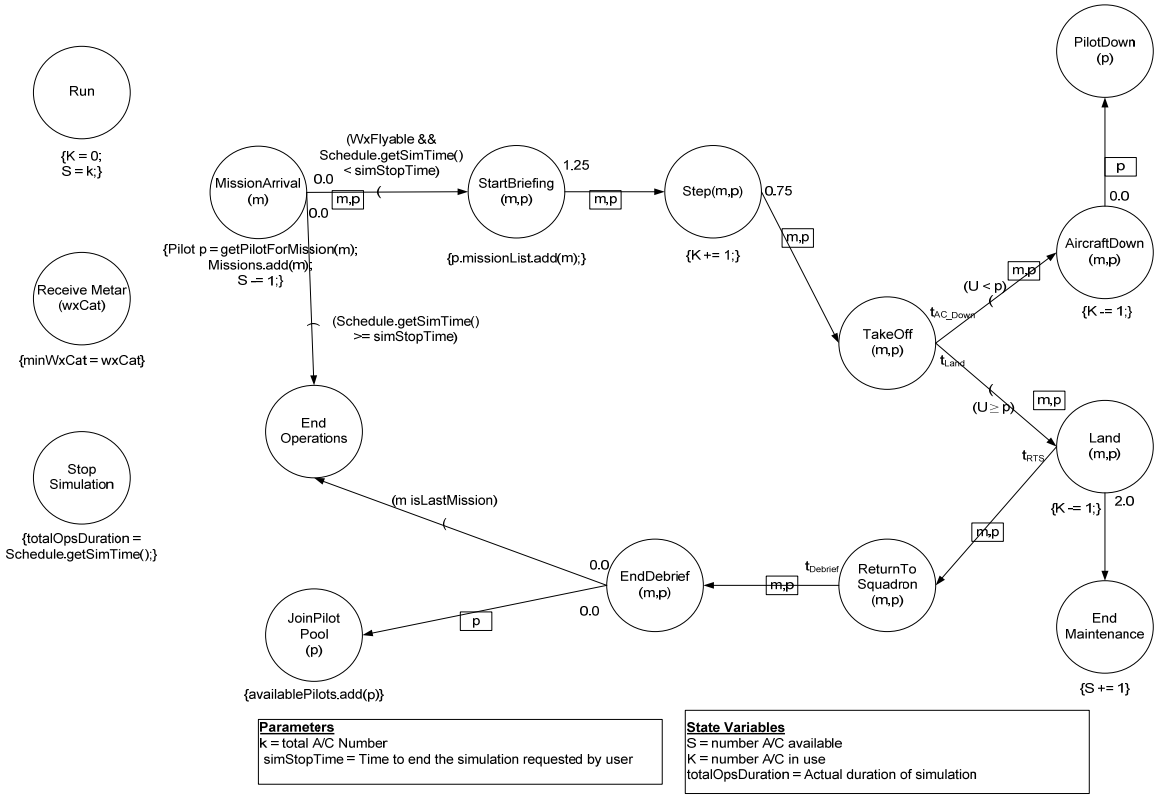


Figure 11. Mission Server Event Graph.

Figure 11 shows the event graph for MissionServer. This module listens to the MissionCreator, DutySchedulerBFO, and DutySchedulerBWOC modules. The purpose of this module is to define the features of pilots required for the missions and execute events related to these pilots. The MissionArrival event is scheduled by the Arrival event in the MissionCreator module. The MissionArrival event receives the mission object created by MissionCreator. This method assigns pilots to missions.

The doMissionArrival() method calls a method named getPilotForMission(). This method uses the following logic to retrieve pilots for the mission from Squadron module: The Squadron module may lend two types of pilots to the MissionServer module. If Squadron has a pilot that can satisfy the minimum requirements of the mission, then it will give this pilot to the MissionServer. If Squadron does not readily have a pilot with the necessary qualifications for the mission, then it will create a pilot with desired qualifications. At the beginning of the simulation, there are no pilots in the squadron. The qualifications that the MissionServer module needs to define are the flight position, IFR weather category, and the LANTIRN category of the pilot. This simulation tries to form a balanced distribution of pilot manpower force. For this purpose, the getPilotForMission() method defines two different sets of qualifications for the pilot. The first set consists of minimum necessary qualifications. The second set consists of the desired set of qualifications. For example, the simulation needs a wingman pilot for a LANTIRN mission. The Squadron module will first search through the wingmen that it has. If there is a wingman at hand that also satisfies the IFR weather category and LANTIRN category requirements, it will return that pilot. If not, it will search through two-ship leads. If unsuccessful, then it will search through four-ship leads. If it finds a four-ship lead that can fly in this wingman position, then it will give it to the MissionServer module. If it is unsuccessful, i.e., there are no pilots readily available that satisfy the minimum requirements, it will create a wingman and return it to the MissionServer module. The IFR weather category and LANTIRN category of that wingman will be designated by the desired set of qualifications.

Weather conditions around the take-off and landing airfield dictate the IFR category of the pilots. The MissionServer module listens to the WeatherStation module via a listener pattern. At the beginning of every hour, the WeatherStation module disseminates the minimum category required for the pilots who will fly in this hour. Pilots who have categories lower than the minimum required category will not fly. The doReceiveMetar() method receives the minimum category value.

After retrieving the pilot for the mission, the doMissionArrival() method starts scheduling follow-on events for the pilots. The doMissionArrival() will schedule the same event flow for every pilot in the mission flight. In other words, if a mission requires four aircraft, the MissionArrival event will schedule four StartBriefing events. This method will also reduce the number of aircraft available by one.

The execution of the simulation does not stop at a predetermined time. In real life, operations on base end when the last aircraft lands. In the simulation model, the desired end time of operations is determined by an input factor. After the end time of operations is due, the simulation checks the debriefing of the last landing. After the last debriefing, the MissionServer module schedules the EndOperations event. DutySchedulerBFO and DutySchedulerBWOC listen for that event. Upon hearing that event, they also end their operations and both fire the StopSimulation event. The MissionServer module listens to the DutySchedulerBFO and the DutySchedulerBWOC modules for that event. When it hears these events, it stores the current simulation time. The simulation stops after two StopSimulation event fires.

If current weather conditions allow flight operations and the simulation end time has not yet been reached, MissionArrival will schedule the StartBriefing event. The doStartBriefing() method will add the mission to the mission list of the pilot. The StartBriefing event takes place two hours before takeoff. This event takes approximately 1 hour and 15 minutes. StartBriefing schedules a Step event at that time. Step means leaving the squadron and arriving to the flight line. The doStep() method increases the number of aircraft in use by one. The doStep() method schedules the doTakeOff() 45 minutes later. This 45-minute value is a fixed value for all flights.

In the doTakeOff() method, a random number is drawn. If this random number is less than the attrition rate, then the aircraft and the pilot are assumed to be lost due to enemy action. The time delay for this shot down event, t_{AC_Down}, is also determined by a random draw. The doAircraftDown() method decreases

the number of aircraft in use by one. It also schedules a PilotDown event. The Squadron module listens for the PilotDown event. If the random number is above the attrition rate, the doTakeOff() method will invoke the doLand() method. Time delay t_{Land} is the duration between takeoff and land events. The MissionCreator module determines this duration from a user-defined triangle distribution and writes it into the mission object.

The doLand() method invokes two methods. The first method is the doEndMaintenance() method. After an F-16 lands, there is a two-hour maintenance period for routine checks. The aircraft is unavailable for any flight activity until this two-hour maintenance period is over. The doEndMaintenance method increases the number of aircrafts available by one, allowing scheduling of new sorties for the aircraft. The second method that the doLand() method invokes is the doReturnToSquadron() method.

After the aircraft lands, the pilot taxies to the de-arm area for de-arming of weapons systems. He then taxies back to shelter, shuts down the engine, and proceeds to the maintenance debriefing room. After the debriefing is over, he returns to squadron. The MissionCreator module stochastically determines this time duration, t_{RTS}, for this whole process from a user-defined triangle distribution and writes it into the mission object.

After a pilot returns to the squadron building, a debriefing event will take place. The ReturnToSquadron event schedules the EndDebrief event after a time delay of $t_{Debrief}$. This time delay is also drawn from a user-defined triangle distribution and written into the mission object by the MissionCreator module. The doEndDebrief() method invokes the doEndOperations() and doJoinPilotPool() methods. The Squadron module listens for this call to schedule its own JoinPilotPool event. This event adds the pilot into the available pilots list. If this mission was the last mission, all operations will end at the end of debriefing of the mission. The EndDebrief event schedules the EndOperations event in the case this was the last mission.

5. Duty Scheduler Supervisor of Flight

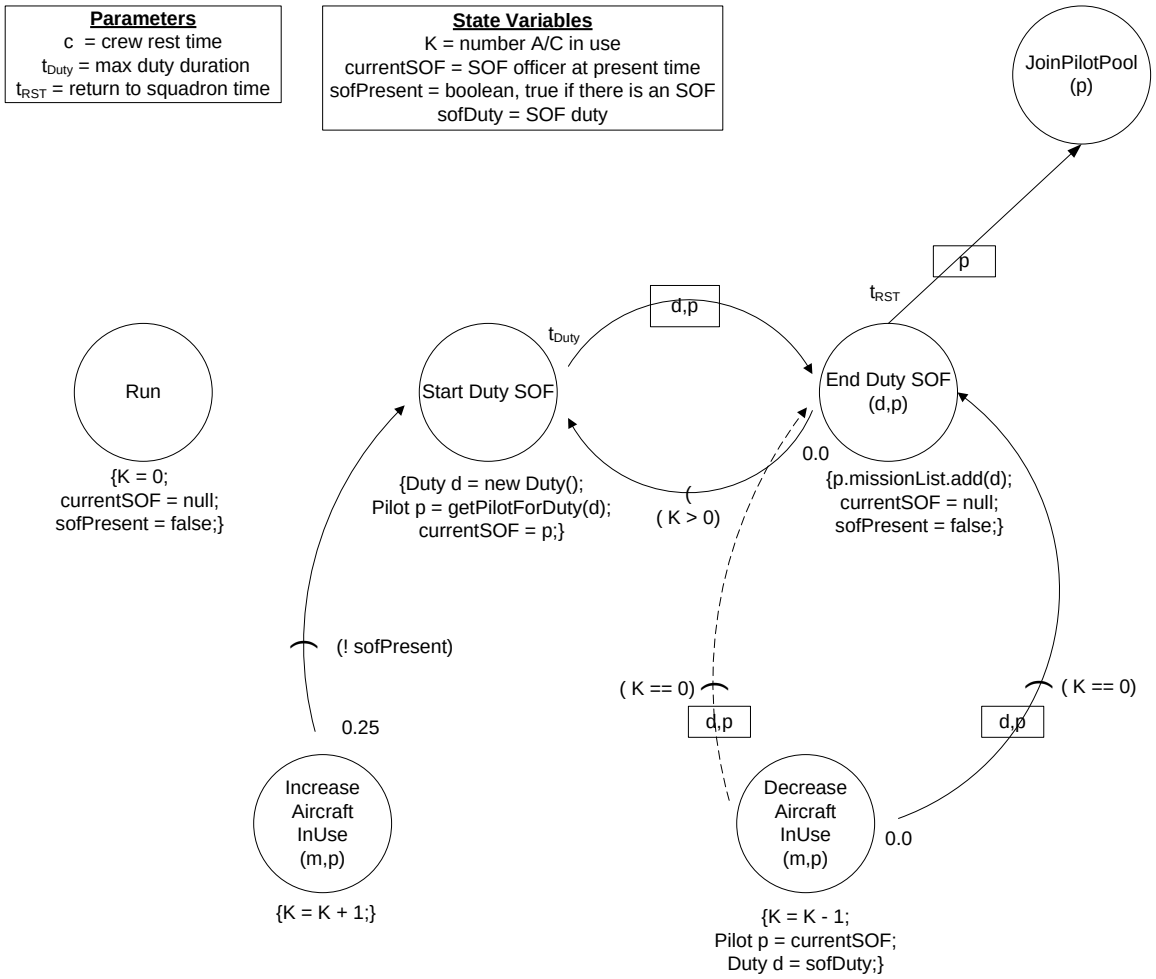


Figure 12. Duty Scheduler Supervisor of Flight Event Graph.

Figure 12 shows the event graph for the DutySchedulerSOF module. The Supervisor of Flight (SOF) is a duty post in the control tower. Whenever there is a flight activity on base, a four-ship lead pilot will be scheduled to carry out this duty. SOF duty starts 30 minutes prior to the first takeoff and ends when the last aircraft lands. There is not a fixed time length for this duty. In the simulation, the maximum time limit for a pilot for this duty is eight hours. If the 8-hour limit is due, another pilot will replace the current SOF. However, if the flight activities stop before the 8-hour limit, the simulation will cease the duty and send the pilot back to squadron.

This module is connected to the MissionServer module with an adapter. When the Step event in the MissionServer module is executed, it invokes the IncreaseAircraftInUse event in the DutySchedulerSOF module. The Land event in the MissionServer module invokes the DecreaseAircraftInUse event in the DutySchedulerSOF module. The Step event in the MissionServer model executes 45 minutes prior to takeoff. It then fires the IncreaseAircraftInUse event, which in turn schedules the StartDutySOF event 15 minutes later if there is not already a SOF in the tower.

The StartDutySOF event retrieves a four-ship pilot from the Squadron module and schedules the EndDutySOF event for the time delay of t_{Duty}. This time delay is eight hours. If there is a continuous flight activity of eight hours or more, the EndDutySOF will execute, send the SOF back to squadron, and schedule another StartDutySOF event immediately. When aircraft land, the Land event in the MissionServer module will execute, and it will invoke the doDecreaseAircraftInUse() method in the DutySchedulerSOF module. If there is no aircraft left flying, the DecreaseAircraftInUse event will interrupt (i.e., cancel) the EndDutySOF event that was scheduled to execute eight hours later and schedule the same event immediately. The EndDutySOF event will end the duty of SOF and will schedule the JoinPilotPool event. The Squadron module listens for that event call. The JoinPilotPool event sends the pilot back to squadron. Since there is no aircraft flying, the EndDutySOF event will not schedule another StartDutySOF event.

6. Duty Scheduler Runway Supervisory Unit

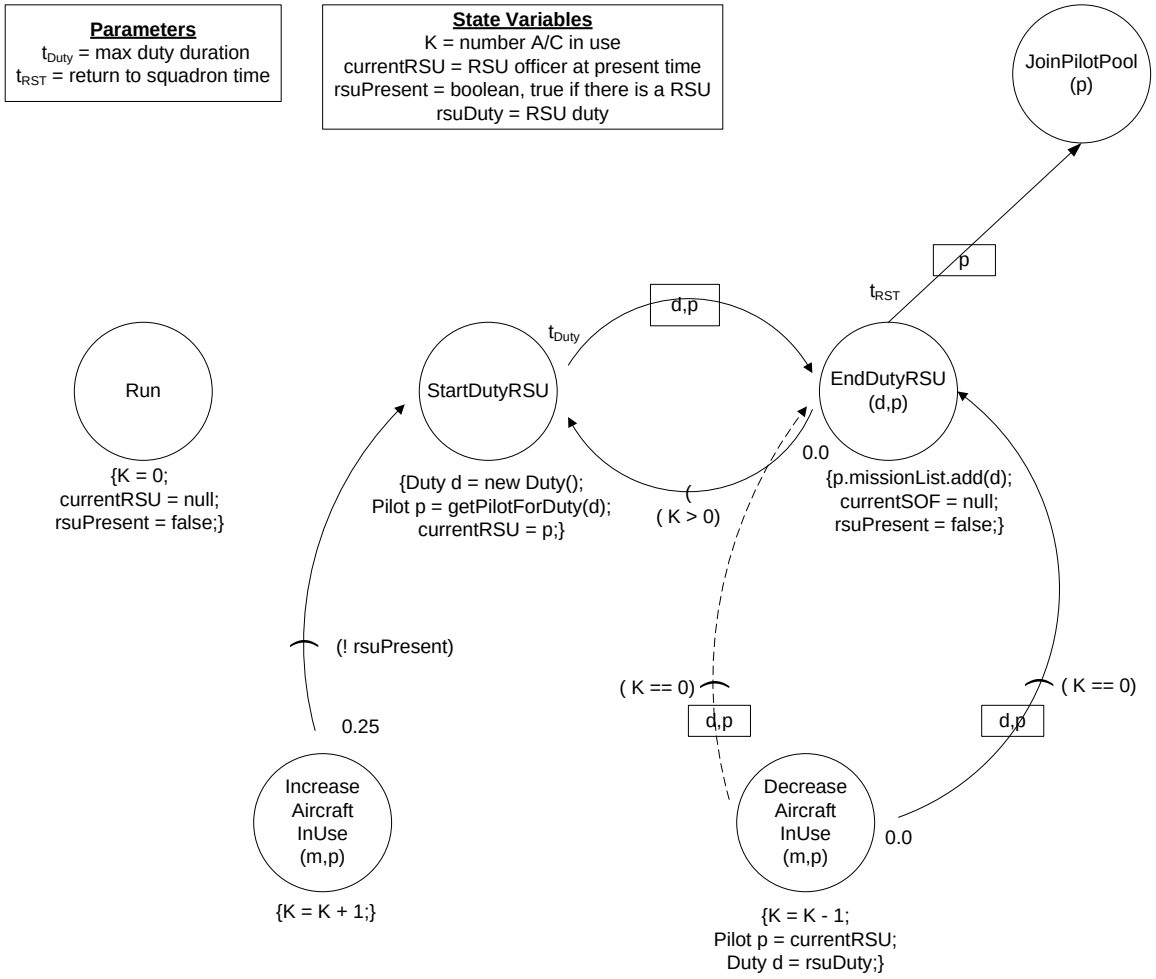


Figure 13. Duty Scheduler Runway Supervisory Unit Event Graph.

Figure 13 shows the event graph for the DutySchedulerRSU module. The Runway Supervisory Unit (RSU) is another flight-related duty post. At the approach at the end of the runway, there is a booth. One pilot stays there and checks the landing aircraft for any irregularities, such as unextended gear, low or high approach angle, high flare, etc. The RSU has the same rules applying to the SOF as far as duty scheduling; the only difference is the pilot's experience level. SOFs have to be four-ship leads, but RSUs can be wingmen and above. The DutySchedulerRSU module has the same event graph methodology with the

DutySchedulerSOF module. The event logic and event flow is the same. The getPilotForDuty() method in the doStartDutyRSU() method will return any pilot who is a wingman or above. It will prefer wingmen. The reader can refer to the explanation of the DutySchedulerSOF for event scheduling flow.

7. Duty Scheduler Base Flight Operations

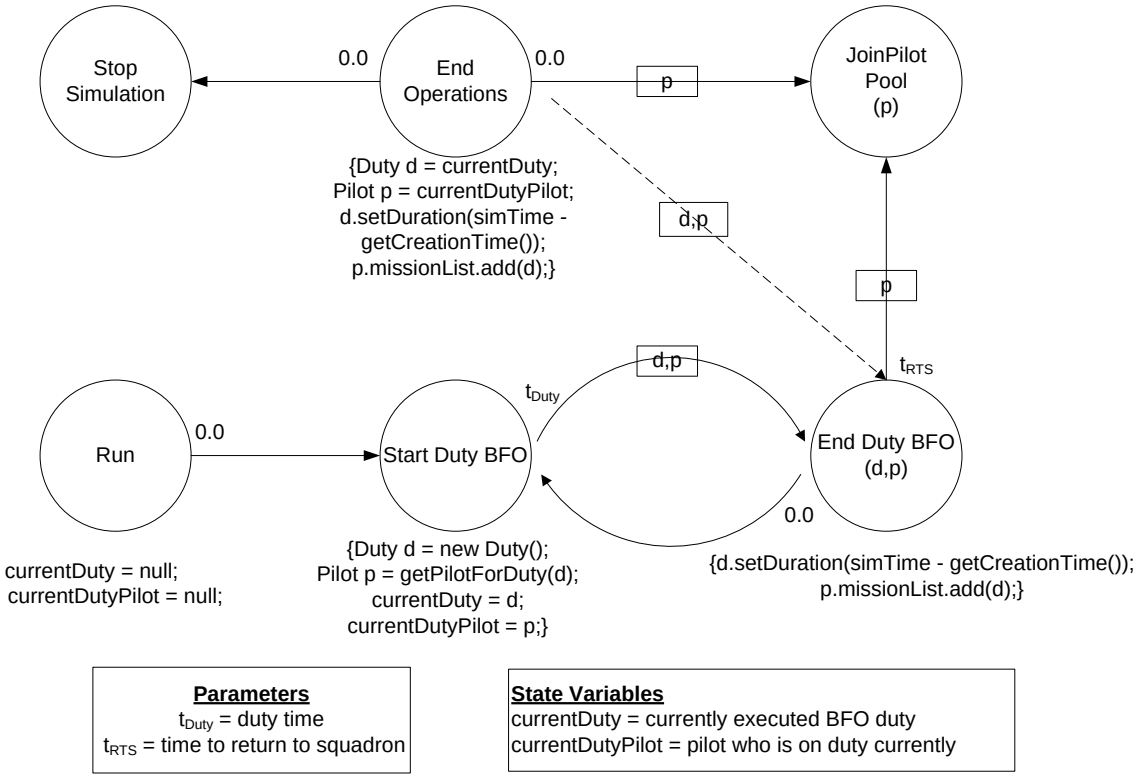


Figure 14. Duty Scheduler Base Flight Operations Event Graph.

Figure 14 shows the event graph for the Base Flight Operations (BFO) duty operations. This is another flight-related duty. All pilots can be scheduled for this duty. However, wingmen and two-ship leads are preferred. The Run event schedules the StartDutyBFO event. The doStartDutyBFO() method creates a new Duty object and assigns a pilot for the duty. The getPilotForDuty() method will retrieve a pilot from the Squadron module. The time delay, t_{Duty}, is currently set to 24 hours. The StartDutyBFO event will schedule the EndDutyBFO event after a time delay of t_{Duty}. The doEndDutyBFO() method records the time duration of duty and adds the duty object to pilot's mission list. It then schedules the

JoinPilotPool event after a time delay of t_{RTS}. This time delay value comes from a user-defined triangle distribution. It denotes the time delay between the pilot's departure from a duty post and his arrival to the squadron. The Squadron module listens for the JoinPilotPool event. It will retrieve the pilot. It also schedules the StartDutyBFO event immediately.

The DutySchedulerBFO module listens to the MissionServer module. The MissionServer module will trigger execution of the EndOperations event. This event ends the BFO duty if the operations end before the 24-hour duty time is over. It cancels the EndDutyBFO event that was scheduled to take place, and sends the pilot back to squadron. It also schedules the StopSimulation event. The simulation ends itself when two StopSimulation events are executed. The second StopSimulation event is scheduled within the DutySchedulerBWOC module.

8. Duty Scheduler Base War Operations Center

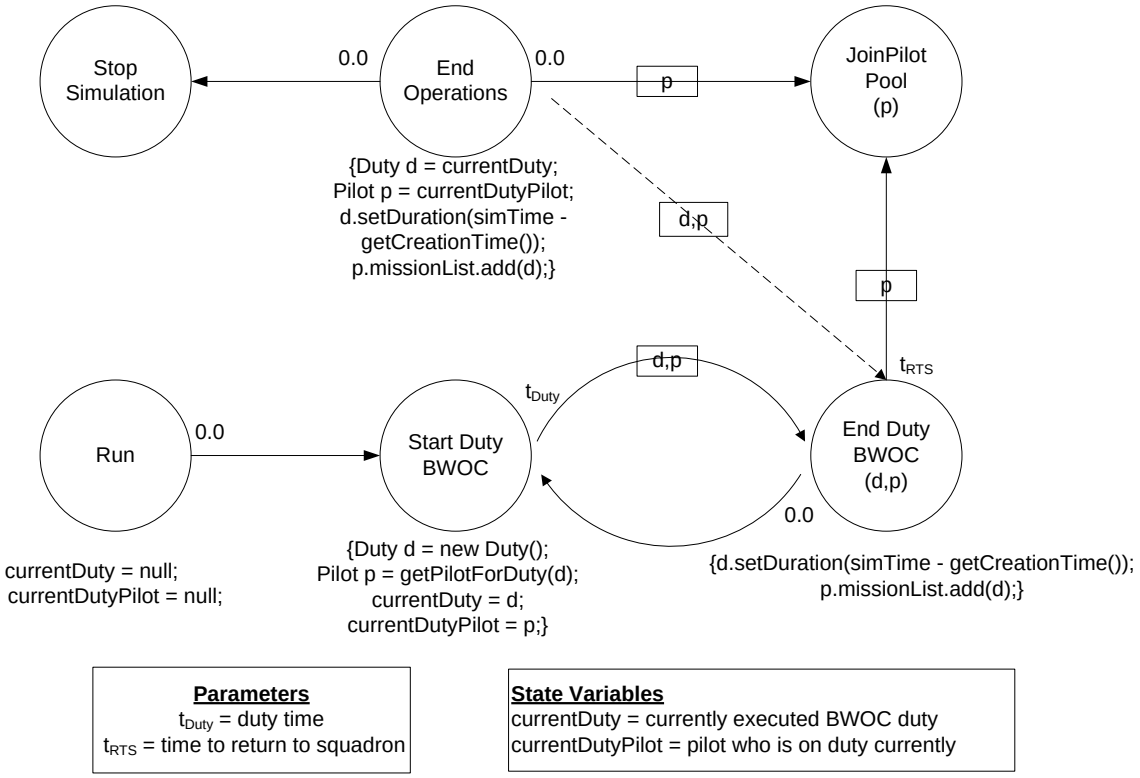


Figure 15. Duty Scheduler Base War Operations Center Event Graph.

Figure 15 shows the event graph for the DutySchedulerBWOC module. The Base War Operations Center (BWOC) is a place where all units on base send a representative to coordinate efforts. This module is the same as the DutySchedulerBFO module. The only difference is the time delay for duty. The t_{Duty} is 12 hours in the simulation. This module listens to the MissionServer module for the scheduling of the EndOperations event. The MissionServer module listens to this module for the StopSimulation event. The Squadron module also listens to this module for the JoinPilotPool event.

9. Squadron

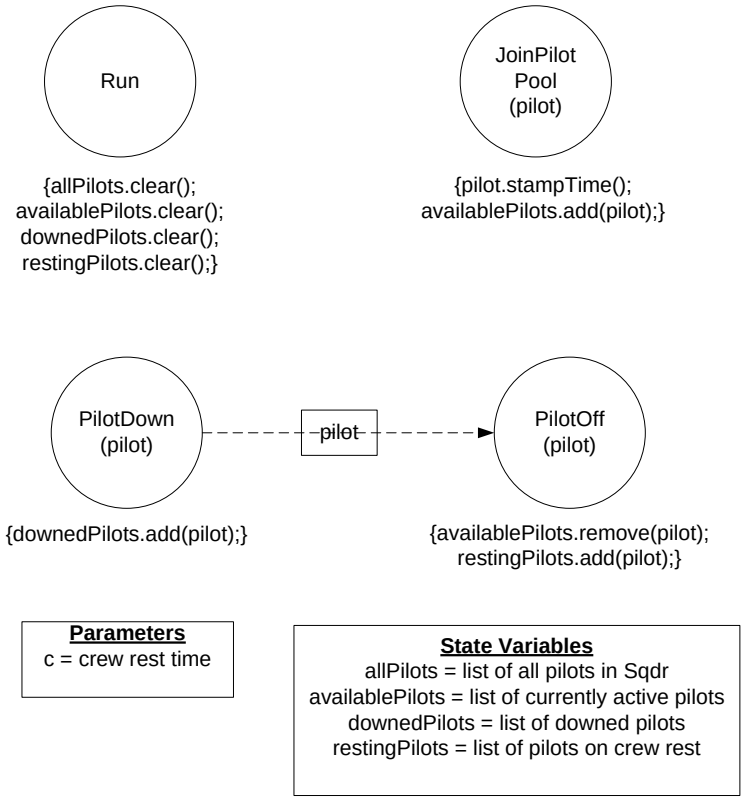


Figure 16. Squadron Event Graph.

Figure 16 shows the event graph for the Squadron module. The Squadron module is responsible for the management of all pilots. The Squadron module listens to the MissionServer, DutySchedulerSOF, DutySchedulerRSU, DutySchedulerBFO, and DutySchedulerBWOC modules to hear the calls for the

events PilotDown, PilotOff, and JoinPilotPool. The Squadron module manages four lists: allPilots, availablePilots, downedPilots, and restingPilots. The allPilots list keeps a copy of all the pilot objects that were created. The availablePilots list keeps the currently active pilots, i.e., pilots who are not on crew rest time. The downedPilots list keeps the pilot objects that were lost to enemy fire. The restingPilots list keeps the pilots who are currently on their crew rest period. The crew rest time is normally twelve hours uninterrupted. However, in this simulation, the user can define a low and a high value for the crew rest period, and vary these to see how the crew rest period length affects the outcome.

When a module needs a pilot, it calls the retrievePilot() method in the Squadron module. The caller needs to supply two sets of categories: minimum required categories for the mission or duty, and desired categories. The Squadron module first searches the availablePilots list. If there is a pilot who can satisfy the minimum requirements with enough time left to complete the mission, then the Squadron will give this pilot to the caller module. If the search is unsuccessful, the Squadron module searches the restingPilots list. If it finds a pilot satisfying the minimum categories who has rested for at least the defined crew rest time, it removes this pilot from the restingPilots list and gives it to the caller module. If these searches are unsuccessful, the Squadron module creates a new Pilot object using the desired categories that the caller module relayed. The following is an example to clarify the logic: The MissionServer model needs to schedule a pilot for a 4.5-hour mission. Weather conditions do not allow IFR category-3 pilots to fly. Therefore, the minimum IFR category is category-2. In addition, this is a LANTIRN mission. The minimum LANTIRN category is category-3, and the pilot will fly in the number-two position in the formation. Therefore, the minimum flight position is wingman. These categories constitute the minimum categories. The Squadron module searches the availablePilots and the restingPilots lists using these minimum categories. If the module finds a pilot who can satisfy all these requirements, it will return that pilot. If the pilot is in the availablePilots list, he also should have enough time left to complete the mission

before his crew rest period begins. As a result, the module can assign a four-ship lead pilot to a wingman position. If the search is unsuccessful, then the Squadron module will create a new pilot. The MissionServer module will define a desired set of categories after some stochastic processes. In this case, the LANTIRN category of the pilot could be Cat-3, Cat-2, or Cat-1. The MissionServer module will determine this category value after a random draw. Let us assume that the result is Cat-2. The MissionServer module also needs to determine the desired weather category of the pilot. A wingman could be either IFR Cat-3 or IFR Cat-2. Since the weather minimums are below Cat-3 minimums, the IFR category will be Cat-2. Therefore, the result for the desired qualifications is wingman, LANTIRN Cat-2, and IFR Cat-2. The Squadron module will create a pilot with these category and flight position values and give him to the MissionServer module.

Whenever a new pilot is created, a PilotOff event is scheduled for this pilot at the end of his up time. Up time is 24 hours minus the crew rest time. The PilotOff event removes the pilot from availablePilots list and puts him into the restingPilots list. If a pilot is lost due to enemy fire, the PilotDown event will cancel his PilotOff event and put him into the downedPilots list. The modules that use pilot objects return them to the Squadron module by the JoinPilotPool event. The Squadron module listens for this event. The returned pilot object is placed back into available pilots list. The doRun() method clears all the lists at the beginning of the program.

10. Weather Station

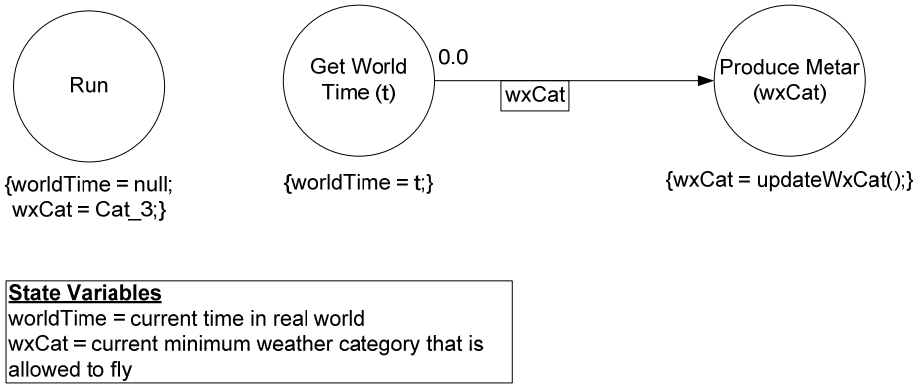


Figure 17. Weather Station Event Graph.

Figure 17 shows the event graph for the WeatherStation module. The WeatherStation module is responsible for producing a minimum IFR weather category value hourly. The MissionServer module is connected to this module and receives the minimum weather category value by its doReceiveMetar() method. The MissionServer module does not schedule pilots whose IFR category is below the minimum weather category.

The WeatherStation module is connected to the SimulationCalendar class via an adapter pattern. At the beginning of every hour, its doGetWorldTime() method is invoked by the doAnnounceWorldTime() method of the SimulationCalendar class. The GetWorldTime event then schedules the ProduceMetar event. The ProduceMetar event receives the current minimum category as an argument. This is because the methods that are used in the adapter pattern have to have the same parameter list. The ProduceMetar event relays the current weather category value to the ReceiveMetar event by using the current weather category as an argument. The GetWorldTime event, in turn, has to use the same argument to call the ProduceMetar event. The ProduceMetar event then updates the current weather category by using Markov chains. Chapter III explains the construction of the weather model and production of hourly weather categories in detail.

C. ASSUMPTIONS

1. Pilots who join the squadron as their first assignment after F-16 training have 120 flight hours as a primary pilot. They are wingman, IFR Cat-3 pilots with no LANTIRN or MANTIRN category.

2. Pilots who join the squadron as their second assignment are four-ship leads with IFR Cat-1 weather category. Three out of four pilots are LANTIRN Cat-1. One out of four pilots has no previous LANTIRN training.

3. Pilots fly 150 hours per year.

4. After spending one training year in the squadron, pilots can start receiving MANTIRN category training.

5. After spending two years in the squadron, pilots can start receiving LANTIRN category training.

6. Pilots who reach the 500-hour flight mark are upgraded to two-ship lead position and IFR category-2.

7. Pilots who reach the 750-hour flight mark are upgraded to four-ship lead position and IFR category-1.

8. MANTIRN Combat Readiness (MCR) training takes 15 days to complete.

9. MANTIRN Cat-4 training takes two months to complete.

10. LANTIRN Combat Readiness (LCR) takes one and a half months to complete.

11. LANTIRN Cat-2 Training takes two months to complete.

12. LANTIRN Cat-1 Training takes three months to complete.

13. Pilots get a new assignment at the end of the fifth year.

The following two figures show the respective timelines for the first assignment pilots and the second assignment pilots. The simulation uses these timelines to determine the categories of the pilots that are to be created stochastically. For example, if the pilot needs to be a LANTIRN pilot, then the simulation will calculate the ratios of times that pilot spends in each category and make a random draw. The result of the random draw determines the category.

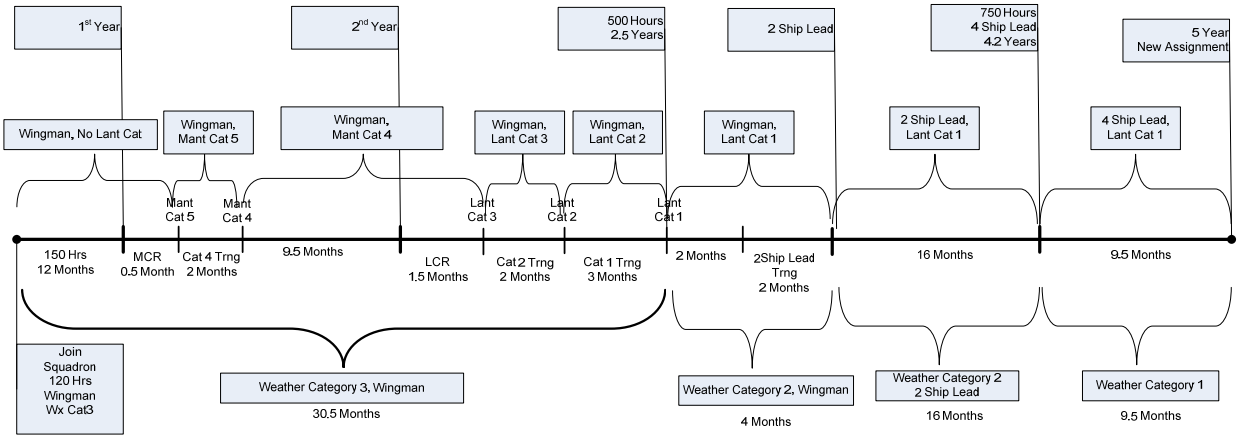


Figure 18. Timeline for the Inexperienced Pilot in the LANTIRN Squadron.

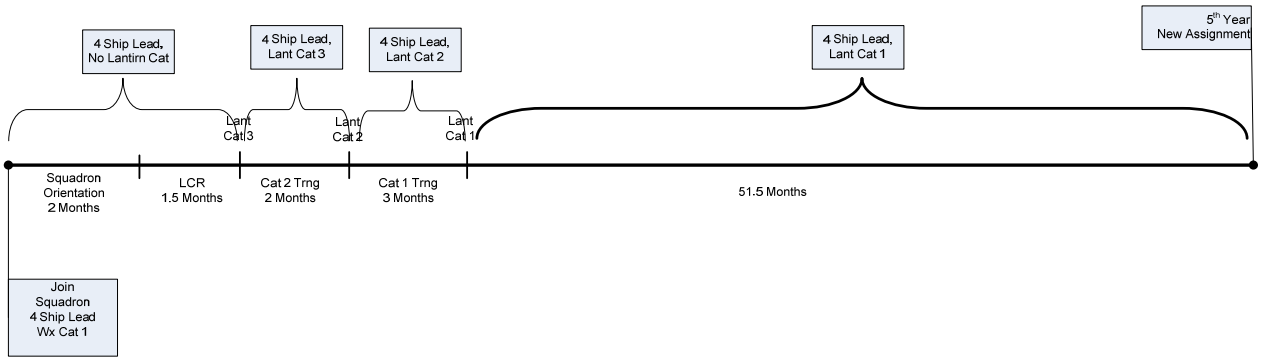


Figure 19. Timeline for the Experienced Pilot in the LANTIRN Squadron.

III. WEATHER MODEL

Weather events affect flight operations to a great extent. There are two situations where weather conditions affect flight operations. The first situation is weather conditions around the takeoff and landing airfield. The second situation is weather conditions in the operations area. The modeling of weather conditions in target area is out of scope of this thesis; therefore, it was not modeled. On the other hand, weather conditions around the takeoff airfield affect mission scheduling and pilot selection. This thesis models and simulates weather conditions around the airfield using a Markov chain.

A. PILOT WEATHER CATEGORIES

There are three levels of pilot weather categories. Pilots start as weather category 3, become weather category 2, and finally are promoted to weather category 1 as they become proficient and accumulate more flight hours. If current weather conditions are below a pilot's weather category minimums, then that pilot will not take off. The squadron scheduler assigns a new pilot with a sufficient category.

Table 2. Weather Ceiling and Visibility Minimums for Each Category.

Weather Category	Limits
I	Ceiling 200 feet or above, visibility 800 meters or above
II	Ceiling 500 feet or above, visibility 2000 meters or above
III	Ceiling 1000 feet or above, visibility 3000 meters or above

1. Weather Category 3

When pilots first join operational squadrons after F-16 training, they have the rating as weather category-three pilots. This is the lowest level among

weather categories. The minimum required ceiling is 1000 feet, and the minimum required visibility is 3 kilometers. If ceiling and/or visibility are below these values, then weather category-three rated pilots do not fly.

2. Weather Category 2

When a pilot accumulates 500 flight hours as a first pilot and accumulates at least 100 flight hours in a primary aircraft type, he is promoted to be a weather category-two pilot. The minimum required ceiling is 500 feet, and the minimum required visibility is two kilometers. If ceiling and/or visibility are below these values, then weather category-two rated pilots do not fly.

3. Weather Category 1

When a pilot accumulates 750 flight hours as a first pilot and accumulates at least 150 flight hours in primary aircraft type, he is promoted to be a weather category-one pilot. The minimum required ceiling is 200 feet, and the minimum required visibility is 800 meters. If ceiling and/or visibility are below these values, then flight operations stop in this airfield. There will be no takeoffs until weather conditions get favorable again.

B. METAR REPORTS

“An aviation routine weather report, or METAR, is an observation of current surface weather reported in a standard international format.” (FAA-H-8083-25, 2003). Weather offices located in airfields prepare and broadcast routine hourly METAR reports. Pilots use these reports for briefing and flight planning. METAR reports contain current observed weather conditions around the station and might contain information about other important weather events that are expected to happen within next hour. Decision makers in squadrons (Squadron Commander, Training Officer, and Scheduler) use METAR reports to make changes in flight programs if necessary. For example, if the latest METAR report gives the cloud ceiling as 750 feet, then weather category-three pilots will

not be scheduled for the upcoming missions until the cloud base increases above 1000 feet. An example of METAR report from Balikesir Air Force Base dated January 27, 2007, is as follows: “METAR LTBF 272250Z 33010KT 9999 -SHRA SCT035 BKN100 08/08 Q1008 NOSIG RMK RWY18 31008KT.”

- METAR indicates that following report is a standard hourly report.
- LTBF is the four-letter ICAO (International Civil Aviation Organization) identifier for Balikesir Air Force Base.
- 272250Z indicates that the day of month is 27, and time of day is 2250 Zulu (Greenwich Mean Time).
- 33010KT indicates wind direction is 330 degrees and wind velocity is 10 Knots.
- 9999 indicates current prevailing visibility is 10 kilometers or better.
- -SHRA indicates that there is a light-intensity rain shower.
- SCT035 BKN100, as a group, indicates the amount of cloud cover and height of cloud base from ground level in feet. There are four identifiers for cloud layers. FEW means 1/8 to 2/8 of the sky is obscured. SCT (scattered) means 3/8 to 4/8 of the sky is obscured. BKN (broken) means 5/8 to 7/8 of the sky is obscured. OVC (overcast) means 8/8 of the sky is covered. The cloud height data next to cloud layer is multiplied by 100 to get actual cloud base in feet. Only BKN and OVC layers constitute a sky cover. FEW and SCT do not restrict flight operations no matter how low the ceiling is. The group above indicates that first cloud layer is scattered and the cloud base is at 3500 feet. The second cloud layer is broken and the cloud base is at 10000 feet.
- 08/08 indicates that temperature is 8 degrees Celsius and the dew point is also 8 degrees Celsius.

- Q1008 indicates that current barometric pressure, extrapolated to sea level, is 1008 millibars.
- NOSIG is an example of a trend forecast appended to the end. It indicates there is not a significant weather event expected to occr within the next hour.
- RMK indicates remarks section, and gives various information. This section might not appear. In the above example, it says that the active runway direction is 18 degrees and wind is 310 degrees at 8 knots. This remark appeared because this is a tailwind landing condition.

Further information on METAR report formats is in the Appendix.

C. CONSTRUCTION OF WEATHER MODEL

After a telephone conversation, Mr. Ozcan at the Balikesir Air Base weather station mailed the complete METAR reports of year 2007 (Ozcan, 8 January 2008). Data consist of a total of 8,759 hourly METAR reports arranged in time order for the whole year. A separate Java program developed by the researchers decoded METAR reports. Every METAR report corresponds to one of the four categories. If the ceiling is above 1,000 feet and visibility is three kilometers or better, there is no restriction for flight operations. Weather category-three pilots, weather category-two pilots, and weather category-one pilots can fly. The corresponding designation for this hour is “Category Three.” If conditions are such that visibility is above two kilometers but it is below three kilometers, and the ceiling is above 500 feet but it is below 1,000 feet, then only weather category-two pilots and weather category-one pilots can fly. The designation for this hour is “Category Two.” If visibility is above 800 meters but it is below two kilometers, and the ceiling is above 200 feet but it is below 500 feet, then only weather category-one pilots can fly. The designation for this hour is “Category 1.” If weather conditions do not allow safe flight operations, or visibility is below 800

meters or the ceiling is below 200 feet, then the designation for this hour is "Category 0." The simulation assumes that the weather condition designation does not change until the arrival of next hour's report. Then, the program produced a comma-separated values (.csv) file containing a designation as described above for each of 8,759 hourly METAR reports. The resultant file is a time series that the researchers further analyzed in JMP software to construct a Markov chain weather model. After importing the file into JMP, a lag-1 column is created and added to the table. This column has the lagged values of weather designations. It is possible to form the lagged values by taking the value of the previous hour and assigning it to this hour. For analysis purposes, numeric values are assigned to each of four categories. The numeric values are "0" for Category-0, "1" for Category-1, "2" for Category-2, and "3" for Category-3. Designations used for each hourly METAR report and lagged values are shown in Figure 20.

) - [outputFinal 2007.JMP] - [real world data 2007 with lagged values]

ols View Window Help

real world data 2007 with lagged va

	Count	Year	Month	Day	Hour	Min Wx Cat to Fly	Lag Min Wx Cat	Numeric Wx Cat	Numeric Lag Wx Cat
1	1	2007	1	1	0	Weather Cat 3		3	.
2	2	2007	1	1	1	Weather Cat 3	Weather Cat 3	3	3
3	3	2007	1	1	2	Weather Cat 3	Weather Cat 3	3	3
4	4	2007	1	1	3	Weather Cat 3	Weather Cat 3	3	3
5	5	2007	1	1	4	Weather Cat 3	Weather Cat 3	3	3
6	6	2007	1	1	5	Wx Not Flyable	Weather Cat 3	0	3
7	7	2007	1	1	6	Wx Not Flyable	Wx Not Flyable	0	0
8	8	2007	1	1	7	Wx Not Flyable	Wx Not Flyable	0	0
9	9	2007	1	1	8	Wx Not Flyable	Wx Not Flyable	0	0
10	10	2007	1	1	9	Wx Not Flyable	Wx Not Flyable	0	0
11	11	2007	1	1	10	Weather Cat 1	Wx Not Flyable	1	0
12	12	2007	1	1	11	Weather Cat 3	Weather Cat 1	3	1
13	13	2007	1	1	12	Weather Cat 3	Weather Cat 3	3	3
14	14	2007	1	1	13	Weather Cat 3	Weather Cat 3	3	3
15	15	2007	1	1	14	Weather Cat 3	Weather Cat 3	3	3
16	16	2007	1	1	15	Weather Cat 3	Weather Cat 3	3	3
17	17	2007	1	1	16	Weather Cat 3	Weather Cat 3	3	3
18	18	2007	1	1	17	Weather Cat 2	Weather Cat 3	2	3
19	19	2007	1	1	18	Weather Cat 2	Weather Cat 2	2	2
20	20	2007	1	1	19	Weather Cat 1	Weather Cat 2	1	2
21	21	2007	1	1	20	Weather Cat 3	Weather Cat 1	3	1
22	22	2007	1	1	21	Weather Cat 1	Weather Cat 3	1	3
23	23	2007	1	1	22	Weather Cat 1	Weather Cat 1	1	1
24	24	2007	1	1	23	Weather Cat 1	Weather Cat 1	1	1
25	25	2007	1	2	0	Weather Cat 1	Weather Cat 1	1	1
26	26	2007	1	2	1	Weather Cat 1	Weather Cat 1	1	1

Figure 20. Hourly METAR Report Designations and Lagged Values.

1. Weather Data as Time Series

One way to model weather events is to use Time Series. This is because the current weather conditions depend on past weather conditions. If it is raining now, the probability of having rain in the next hour is increased. “A time series is a collection of observations made sequentially in time” (Chatfield, 1991). Weather hourly data used in the model form a discrete time series, because observations are taken only at specific times that are evenly spaced.

2. Autoregressive Process

“Suppose that $\{Z_t\}$ is a purely random process with mean zero and variance σ^2_Z. Then a process $\{X_t\}$ is said to be an autoregressive process of order p if

$$X_t = \alpha_1 X_{t-1} + \alpha_2 X_{t-2} + ... + \alpha_p X_{t-p} + Z_t$$

This is rather like a multiple regression model, but X_t is regressed not on independent variables but on past values of X_t; hence the prefix `auto.` An autoregressive process of order p will be abbreviated to an *AR(p)* process.” (Chatfield, 1991)

Figure 21 shows a time series plot of minimum categories pilots need to be allowed to fly.

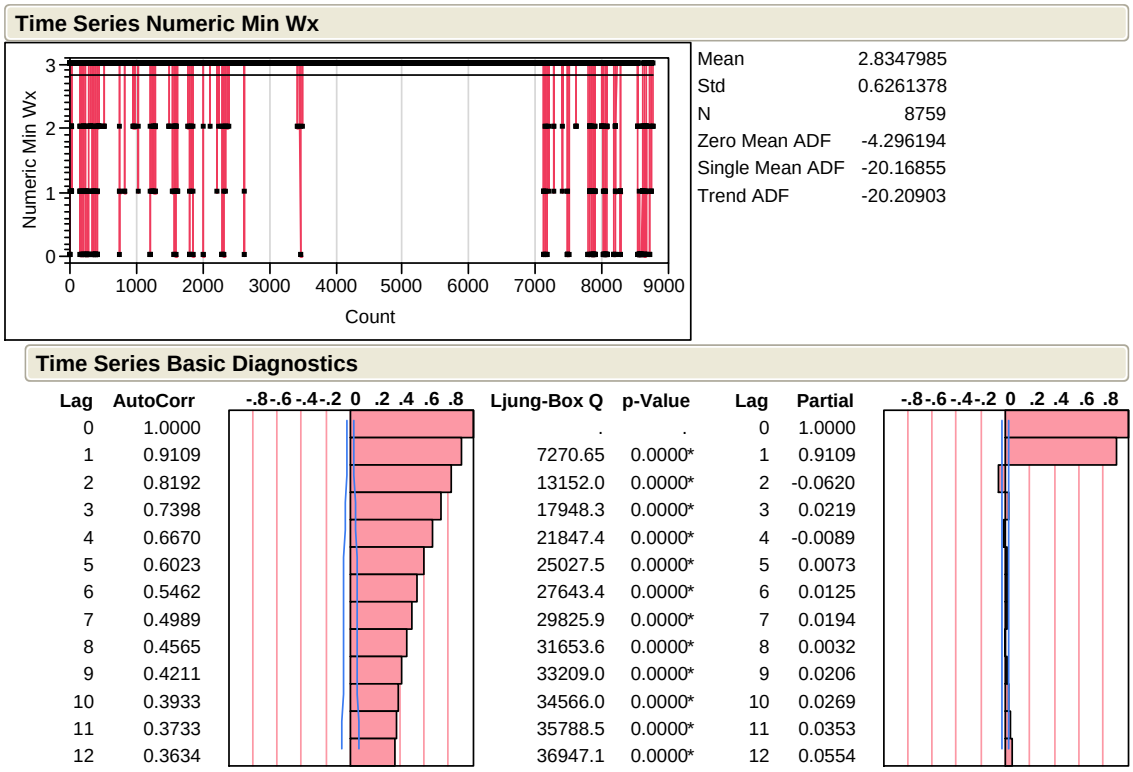


Figure 21. Time Series Report of Hourly Minimum Weather Categories to Fly.

It is obvious from the graph that weather during spring and summer is extremely favorable and does not have any variation. On the other hand, fall and winter weather show fluctuations. For that reason, it is logical to divide the data into two groups. Between May and September, all the METAR reports indicated unrestricted flight operations. Between October and April, weather conditions—and therefore the minimum weather category needed for pilots to be allowed to fly—changed considerably. Figure 22 shows the time series of minimum weather categories that were allowed to fly during this period. In the weather model, it is assumed that flight operations are unrestricted (i.e., weather conditions do not restrict any pilot category from flying) between May and September. The variations within the October–April period are modeled.

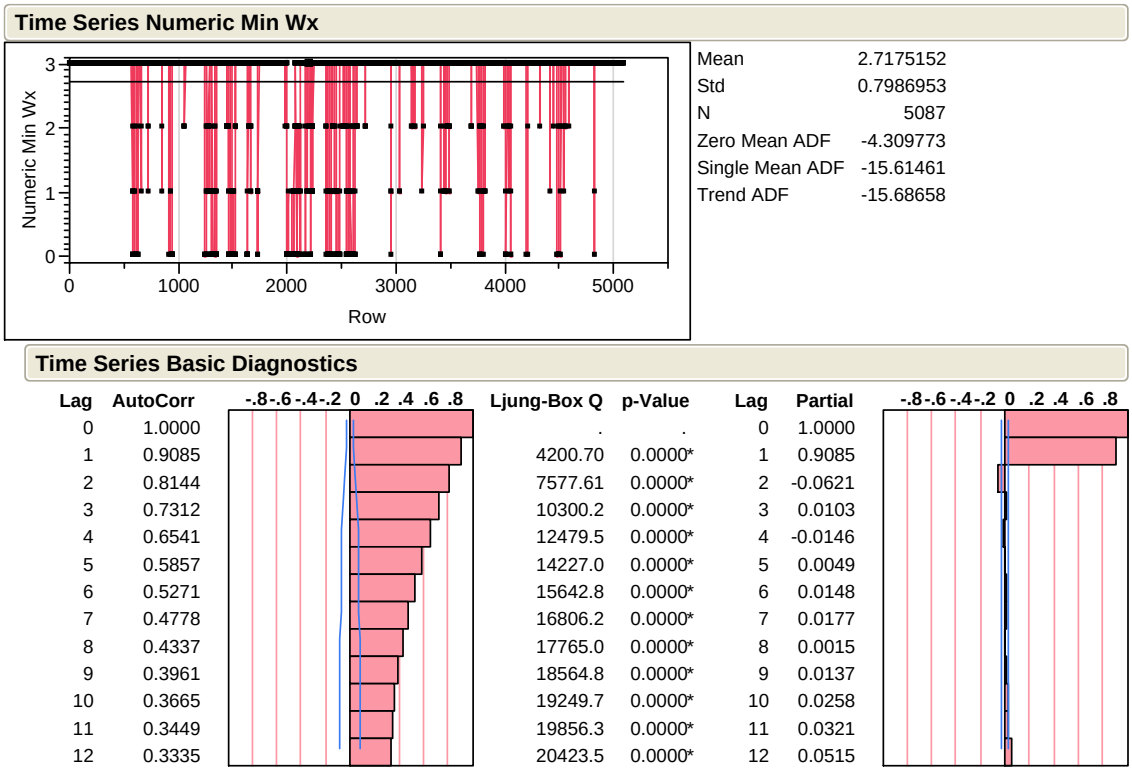


Figure 22. Time Series of Weather Data between October and April.

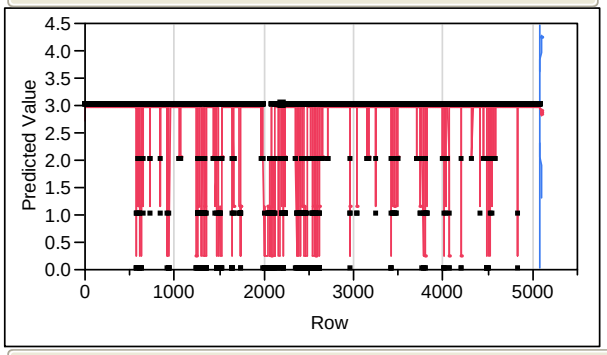
"The plot to the right of the autocorrelation plot is called partial autocorrelation plot. This plot shows the autocorrelations at several lag values after all lower-valued lagged autocorrelations are taken into account." (Sall, Creighton, & Lehman, 2005) The reader can see that there is a spike at lag 1, indicating that model should contain lag-1 term. In order to estimate the parameters of the model, an *AR(1)* model is constructed as in Figure 23.

Model: AR(1)

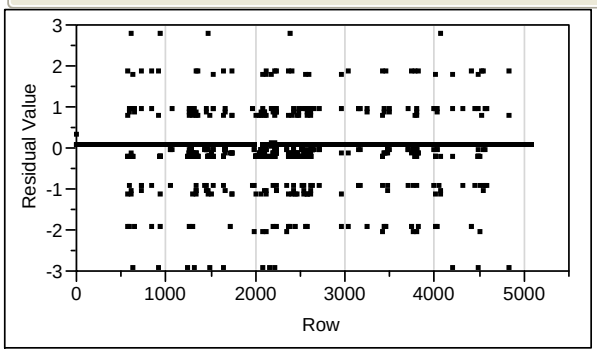
Parameter Estimates

Term	Lag	Estimate	Std Error	t Ratio	Prob>|t|	Constant Estimate
AR1	1	0.9083184	0.0058514	155.23	0.0000*	0.24924672
Intercept	0	2.7186114	0.0507346	53.58	0.0000*	

Forecast



Residuals



Lag	AutoCorr	-.8 -.6 -.4 -.2 0 .2 .4 .6 .8	Ljung-Box Q	p-Value
0	1.0000		.	.
1	0.0565		16.2718	<.0001*
2	-0.0169		17.7225	0.0001*
3	0.0031		17.7730	0.0005*
4	-0.0134		18.6844	0.0009*
5	-0.0231		21.3977	0.0007*
6	-0.0227		24.0214	0.0005*
7	-0.0042		24.1129	0.0011*
8	-0.0128		24.9438	0.0016*
9	-0.0226		27.5451	0.0011*
10	-0.0240		30.4855	0.0007*
11	-0.0363		37.1940	0.0001*
12	-0.0192		39.0732	0.0001*

Lag	Partial	-.8 -.6 -.4 -.2 0 .2 .4 .6 .8
0	1.0000	
1	0.0565	
2	-0.0201	
3	0.0053	
4	-0.0143	
5	-0.0214	
6	-0.0208	
7	-0.0025	
8	-0.0133	
9	-0.0217	
10	-0.0231	
11	-0.0356	
12	-0.0170	

Figure 23. AR(1) Model of Weather Data between October and April.

The lag-1 autoregressive coefficient is 0.9083. The next step is to check the autocorrelation and the partial autocorrelation plot of the residuals. There is no significant autocorrelation or partial autocorrelation. Therefore, it is safe to assume that the *AR(1)* model is adequate.

3. Construction of Discrete-Time Markov Chain Model for Simulation

As the *AR(1)* model showed, the minimum weather category to fly at the current hour can be determined using the previous hour's value. To simulate production of the minimum weather category to fly, a discrete-time Markov chain model is constructed.

In discrete-time Markov chains, the state changes at certain discrete time instances. "The Markov chain is described in terms of its transition probabilities p_{ij}: whenever the state happens to be i, there probability of p_{ij} that the next state is equal to j.

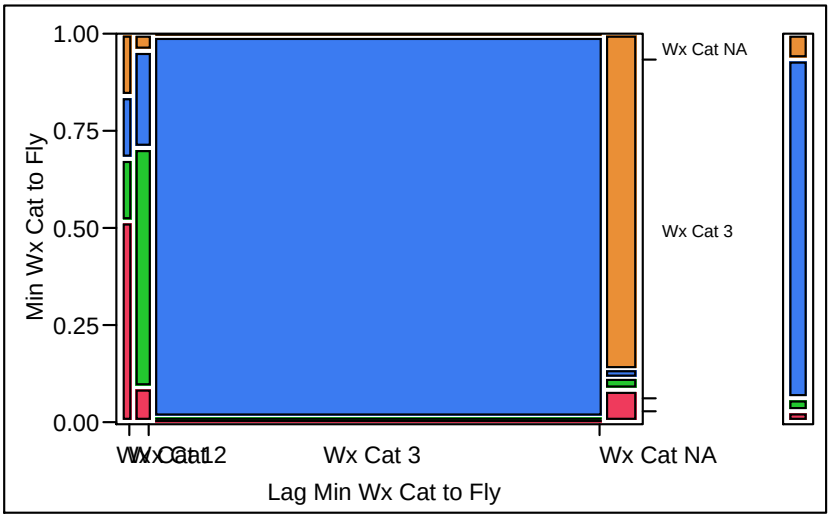
$$p_{ij} = P(X_{n+1} = j \mid X_n = i), \quad i, j \in S$$

The key assumption underlying Markov chains is that the transition probabilities p_{ij} apply whenever state i is visited, no matter what happened in the past, and no matter how state i was reached." (Bertsekas & Tsitlikis, 2002)

To construct a Markov chain model, it is necessary to plot current hour values against lag-1 values. JMP provided the contingency table and probabilities of state transitions between the minimum weather categories to fly (Figure 24.). For example, if the current-hour minimum weather category to fly is Category-3, then probability of staying as Category-3 is 0.9833 in the next hour. If current weather minimum is Category-0 (i.e., weather condition preclude flight operations), then the probability to switch to Category-1 is 0.0841. The simulation uses this resultant Markov chain for weather events modeling. The WeatherStation class handles production of the hourly values of minimum weather category to fly. This produced value affects flight operations and scheduling of pilots to missions.

Contingency Analysis of Min Wx Cat to Fly By Lag Min Wx Cat to Fly

Mosaic Plot



Contingency Table

Min Wx Cat to Fly

Lag Min Wx Cat to Fly	Count Row %	Wx Cat 1	Wx Cat 2	Wx Cat 3	Wx Cat NA	
	Wx Cat 1	75 52.08	23 15.97	23 15.97	23 15.97	144
	Wx Cat 2	17 9.14	115 61.83	46 24.73	8 4.30	186
	Wx Cat 3	25 0.56	37 0.83	4362 98.33	12 0.27	4436
	Wx Cat NA	27 8.41	11 3.43	5 1.56	278 86.60	321
		144	186	4436	321	5087

Figure 24. Mosaic Plot and Contingency Table between Minimum Weather Category to Fly and Lagged Minimum Weather Category to Fly.

4. Goodness of Fit

A goodness-of-fit test can reveal if the values produced by the simulation are consistent with the real world data. There are four categories of minimum weather to fly. These are Category-0 (no flight operations), Category-1 (only weather category-1 pilots can fly), Category-2 (weather category-1 and weather category-2 pilots can fly), and Category-3 (all the pilots in the squadron can fly). Counts of these categories from the real-world data appear in Table 3.

Table 3. Number of Occurrences of Each Category throughout the Whole Year of 2007.

Minimum Category to Fly	Count
Category-0	321
Category-1	144
Category-2	186
Category-3	8108

After constructing the weather model, the simulation was run for 100 years to produce simulation weather output data. Discrete event simulation produces a minimum category for each hour. Pilots who have this minimum category or a better category can fly in this current hour. Simulation produces a new minimum category value at the beginning of the next hour. Table 4 shows the means of counts of categories for 100 years.

Table 4. Mean of Occurrences of Each Weather Category After 100 Years of Simulation.

Minimum Category to Fly	Average Count
Category-0	322.67
Category-1	142.19
Category-2	183.93
Category-3	8109.21

A chi-square goodness-of-fit test is used. “The test statistic, called chi-square (or chi-squared) statistic, is found by adding up the sum of the squares of the deviations between the observed and expected counts:

$$\chi^2 = \sum_{allcells} \frac{(Obs - Exp)^2}{Exp}$$ (De Veaux, Velleman, & Bock, 2005).”

Observed values are the simulation output means for each category. Expected values are the data counts for each category from the real-world data.

$$\chi^2 = \frac{(322.67 - 321)^2}{321} - \frac{(142.19 - 144)^2}{144} - \frac{(183.93 - 186)^2}{186} - \frac{(8109.21 - 8108)^2}{8108}$$

$$\chi^2(3,8760) = 0.054,\ p = 0.99$$

The number of degrees of freedom is 4-1=3 from the n-1 formula, where n is the number of cells. The resultant p-value is 0.99. This result shows us that the distribution of simulated data is consistent with the real-world data.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. EXPERIMENT SETUP, RESULTS AND OUTPUT ANALYSIS

This chapter presents the simulation experiment setup and a guideline for output analysis. Due to the nature of the research question, information for some of the input factors of the simulation was unavailable. To overcome this difficulty, a graphical user interface for the simulation in the Java programming language was used. Future users of the simulation tool can use this GUI to input the correct and valid ranges for the parameters and input factors, choose an experiment design, run the simulation, and get the outputs for further analysis. The reader should note the fact that results that researchers analyze in this chapter do not constitute a valid solution to the research question. Rather, this chapter is a guideline for the analysis of the output that future analysts will retrieve from the simulation with correct inputs. JMP software version 7.0 (SAS Institute Inc., 2007) was used for the analysis in this thesis.

A. GRAPHICAL USER INTERFACE AND EXPERIMENTAL DESIGN

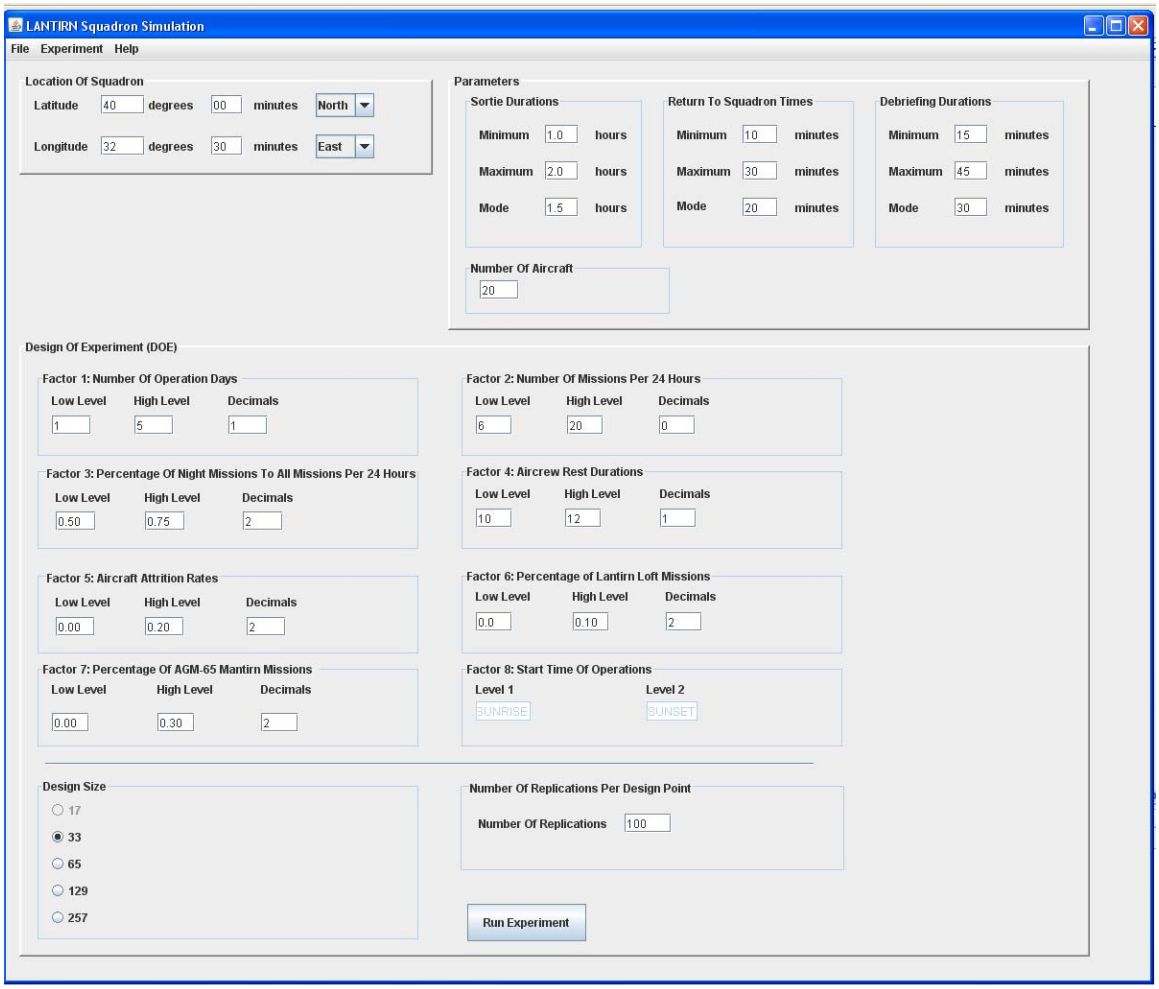


Figure 25. Graphical User Interface of the Simulation.

The user interacts with a GUI to set up the experimental design. There are three panels on the GUI. On the first panel, the user enters the geographic coordinates of the squadron. The second panel is for the entry of fixed parameters. The third panel is dedicated for the experimental design. The user enters the ranges for input factors.

1. Location of the Squadron

The first panel allows the user to enter the location of the squadron. The user enters latitude and longitude as degrees and minutes. The simulation uses

these location values for the sunrise and sunset calculation purpose. Sunrise and sunset times change based on the geographical location and time of year. The simulation uses sunrise and sunset times to designate mission types based on day or night. For example, when a mission is produced, it will be defined as either a LANTIRN or a MANTIRN mission if it is nighttime. If it is daytime, the mission type can be LANTIRN, MANTIRN, or SAT (Surface Attack Tactics).

2. Parameters

The second panel on the GUI is dedicated to the parameters. In this panel, there are four subpanels. The GUI expects the user to specify the parameters that he wishes to use in the simulation. For sortie durations, return to squadron times, and debriefing times, the simulation uses triangular distributions. The reason it uses triangular distribution for these input parameters is the absence of data. There is a lack of real data for these parameters. Using triangular distributions in the absence of the data is one of the heuristics. Minimum and maximum values define an interval for the distribution, and the mode is an estimate of the most likely value (Law, 2007). Sortie durations denote the durations of missions from takeoff to landing. The simulation uses a triangular distribution with the user-defined minimum, maximum, and mode values to produce mission durations stochastically. After landing, pilots taxi to the de-arm area for aircraft de-arm procedures, taxi back to shelter, and go to the maintenance debrief room. After the maintenance debrief, pilots return to squadron. The simulation calls this time amount as “return to squadron times.” The simulation uses entered values for the triangular distribution that produces these “return to squadron times.” It follows a similar approach for debriefing durations after each mission. The fourth subpanel allows the user to enter the initial aircraft number that the squadron has at the beginning of the simulation. This aircraft number is attrited according to user entered attrition rates. Attrition rate is one of the input factors.

3. Design of Experiment

The third panel is the Design of Experiment (DOE) setting place. The simulation uses eight input factors. On seven of them, there are fields for low setting, high setting, and decimal places. There are two fixed levels on the last input factor, which is not user selectable. This input factor is the start time of simulation. Its two levels are sunrise and sunset. The user then selects the number of design points to use. This simulation experiment uses Nearly Orthogonal Latin Hypercube (NOLH) design. NOLH designs have some of the space filling properties of full factorial designs, but requires less sampling (Sanchez, 2006). The simulation produces design points based on the values of the input factors using a Java tool created by Professors Susan Sanchez and Paul Sanchez. The user has the option to choose a design with 33, 65, 129, and 257 design points, based on the designs developed by Cioppa and Lucas (Cioppa & Lucas, 2007). Using a design that has higher number of design points prevents pairwise correlations. "…let *k* denote the number of factors, and let *N*>=*k* denote the number of design points. … Random LH designs have good orthogonality properties if *N* is much larger than *k,* but for smaller designs some factors might have high pairwise correlations*"* (Sanchez, 2006). If the user enters low levels, high levels, and decimals for all factors in such a way that there are at least 17 values between the low and high levels, pairwise correlations will be zero. If that is not the condition, then the user may need to use a larger design to reduce the amount of pairwise correlations. One hundred replications take approximately two seconds to run; 257 design points with 100 replications will take approximately four minutes to run. The total run time for the simulation is not an issue. The user should use larger designs as conditions permit. The next two graphs show the pairwise correlations of the input factors from 33 design points and 257 design points.

Multivariate

Correlations

	# Ops hours	# Missions per day	Ratio of Night missions	Crew rest amount	Attrition rate	% Loft Missions	% Maverick Missions
# Ops hours	1.0000	-0.0303	-0.0007	0.0119	-0.0238	0.0017	0.0218
# Missions per day	-0.0303	1.0000	0.0068	0.0364	-0.0121	0.0096	-0.0377
Ratio of Night missions	-0.0007	0.0068	1.0000	-0.0280	-0.0062	0.0122	0.0254
Crew rest amount	0.0119	0.0364	-0.0280	1.0000	-0.0119	0.0304	-0.0092
Attrition rate	-0.0238	-0.0121	-0.0062	-0.0119	1.0000	-0.0338	0.0138
% Loft Missions	0.0017	0.0096	0.0122	0.0304	-0.0338	1.0000	0.0319
% Maverick Missions	0.0218	-0.0377	0.0254	-0.0092	0.0138	0.0319	1.0000

Scatterplot Matrix

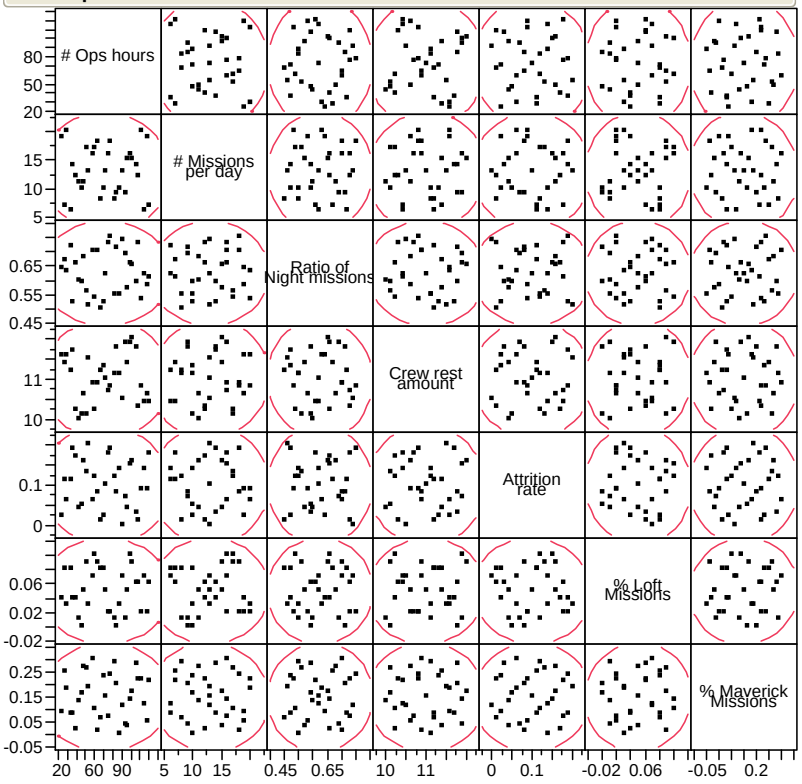


Figure 26. Pairwise Correlation Matrix For a 33 Design Point Simulation Run.

Multivariate

Correlations

	# Ops hours	# Missions per day	Ratio of Night missions	Crew rest amount	Attrition rate	% Loft Missions	% Maverick Missions
# Ops hours	1.0000	-0.0036	-0.0000	-0.0001	-0.0045	0.0044	0.0014
# Missions per day	-0.0036	1.0000	-0.0148	0.0102	-0.0138	0.0033	0.0007
Ratio of Night missions	-0.0000	-0.0148	1.0000	0.0045	0.0003	-0.0082	0.0003
Crew rest amount	-0.0001	0.0102	0.0045	1.0000	-0.0000	0.0064	-0.0091
Attrition rate	-0.0045	-0.0138	0.0003	-0.0000	1.0000	-0.0123	-0.0040
% Loft Missions	0.0044	0.0033	-0.0082	0.0064	-0.0123	1.0000	0.0070
% Maverick Missions	0.0014	0.0007	0.0003	-0.0091	-0.0040	0.0070	1.0000

Scatterplot Matrix

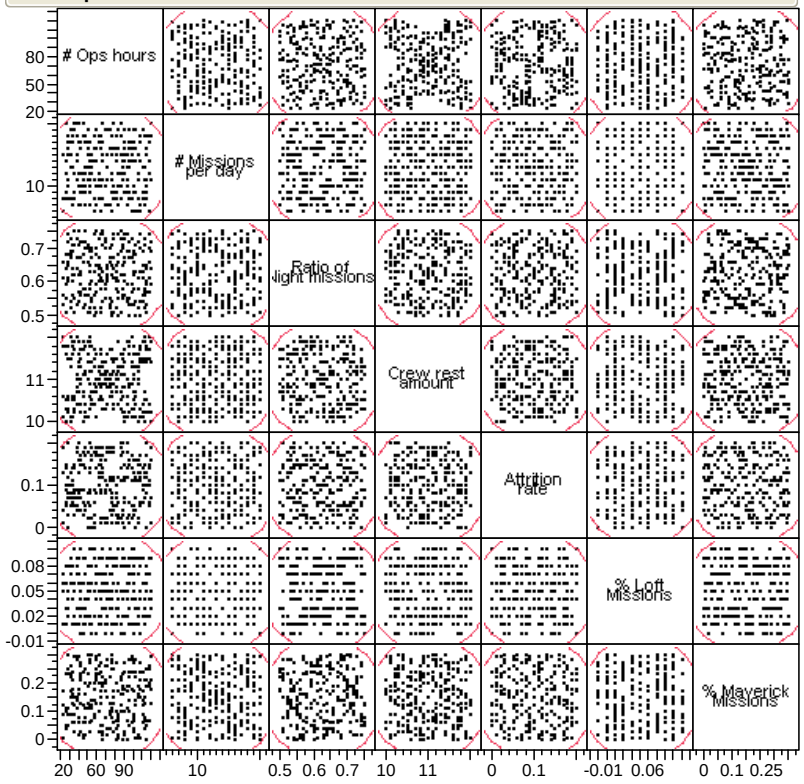


Figure 27. Pairwise Correlation Matrix For a 257 Design Point Simulation Run.

We see that pairwise correlations are smaller in a larger design. They do not zero out, but go through zero.

The last input from the user is the number of replications for each design point. The simulation runs each design point for the specified replication number. Replicating the simulation at each design point is necessary due to the stochastic nature of the simulation. Higher replication numbers are also desirable because they will provide tighter confidence intervals.

4. Input Factors

This section explains the input factors. There are eight input factors. An analyst can define the low level, high level, and decimal places for seven of them. The last input factor has two preset levels.

a. Number of Operation Days

This is the length of the operations in days. The simulation runs for that many days. After reaching the end of this period, the simulation stops producing missions, and the simulation ends after the end of the debriefing of the last mission.

b. Number of Missions per 24 Hours

The simulation uses this input factor to define the arrival rate for the arrival process. The arrival process produces arrival events using a Poisson process. Higher values for this input factor will make the simulation produce more missions.

c. Percentage of Night Missions to All Missions per 24 Hours

The simulation uses a non-stationary Poisson Process. In this process, the arrival rate of the missions changes based on the time of the day. The arrival rate is a function of the time. This factor, together with the previous factor, determines the numbers of day and night missions per 24 hours. The simulation uses different arrival rates for day and night. Higher values for this factor will cause the simulation to produce more night missions than day missions.

d. Aircrew Rest Durations

This input factor defines the time amount that pilots will rest at the end of their work period. The simulation allows pilots to stay active for 24 hours

minus the crew rest amount. At the end of this time, pilots start their crew rest period. During the crew rest period, pilots do not participate in any activity. The crew rest amount is an uninterrupted 12 hours under normal conditions. However, the crew rest period might be adjusted based on the operational needs. The simulation allows analysts to change this crew rest duration between a low and high value to see its effect on the output.

e. Aircraft Attrition Rates

This input factor defines the low and high levels of the attrition rates. The simulation uses stochastic process to decide whether an aircraft is lost during a mission or not. When an aircraft is lost, the simulation lowers the number of aircrafts and pilots by one. It assumes that aircrafts and pilots are not replaced. Low attrition rates equates to an enemy that is less capable. High attrition rates equates to an enemy that is more capable.

f. Percent of LANTIRN Loft Missions

There are two restrictions regarding the loft mission profiles. LANTIRN category-3 pilots cannot fly day or night loft missions. LANTIRN category-2 pilots cannot fly night loft missions. This input factor defines the ratio of LANTIRN missions with a loft attack profile to all LANTIRN missions. The simulation stochastically decides whether a specific LANTIRN mission has a loft attack profile by generating a Bernoulli random variable.

g. Percent of AGM-65 MANTIRN Missions

AGM-65 is a fire-and-forget type air-to-ground missile. If a MANTIRN mission has an AGM-65 weapon load, MANTIRN category-5 pilots cannot fly that mission. The simulation designates a certain portion of MANTIRN missions as AGM-65 missions based on the ratio defined by this input factor. This is also determined stochastically.

h. Start Time of Operations

This input factor has two predefined values. An analyst cannot enter other values. These two levels are sunrise and sunset. Flight operations at the beginning of the simulation start at either sunrise or at sunset.

B. RUN OF EXPERIMENT

After the analyst completes the entry of parameters and input factors, he can run the simulation. The user can either start the simulation run by “Run Experiment” button or “Experiment/Run” menu item. The simulation reads the parameters, the design size, and input factor settings to form the NOLH design. A NOLH design with the user-selected size is formed and filled with values from the input factors. The simulation takes the replication value and runs the simulation at each design point with a user-defined number of replications. For example, if the replication size is 100, then each design point runs 100 times. At the end of each replication cycle, the simulation calculates the mean values for the output MOEs and writes them to a comma-separated file together with input factors. The outputs are the actual run length of simulation in hours, the number of missions flown during the simulation, the number of all pilots needed for the operations, the number of 4-ship leads, the number of 2-ship leads, the number of wingmen, the number of pilots for each LANTIRN and MANTIRN categories (LANTIRN Cat-NA, LANTIRN Cat-1, LANTIRN Cat-2, LANTIRN Cat-3, MANTIRN Cat-4, and MANTIRN Cat-5), and the number of pilots for each weather category (weather Cat-1, Cat-2, and Cat-3). The simulation creates a folder named “Squadron Simulation Output Files” under the “C” drive. After the experiment run is over, the simulation creates a file with “csv” extension. The name of the file is “Generic Squadron Model Outputs.csv.” If the file is already there, then it is overwritten. The simulation ends itself automatically after each run.

C. RESULTS AND ANALYSIS

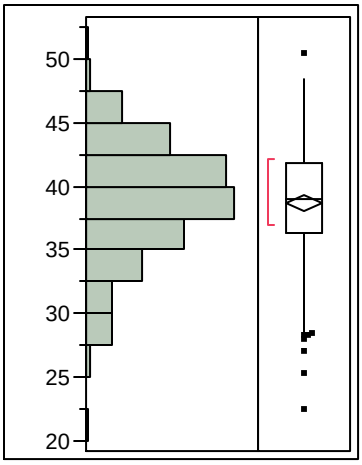
This section will describe some of the analysis that analyst can execute after running the simulation and getting the results. After the analysts runs the simulation, he will find the related output file under “C:/Squadron Simulation Output Files” folder. The main purpose of this simulation tool is to provide a base number for required pilot force and composition of this pilot force. The analysis in this section is based on the results of an experiment with 257 design points and 100 replications for each design point. Since the valid input ranges for parameters and factors are unknown to the authors, arbitrary values are used. Values that were used for the following analyses are on Figure 25. After the simulation run is over, the resultant data table was imported into JMP software for further analysis. The following section gives a general guide that the future analyst can use in his analysis with the results.

1. Basic Statistics

The first statistics that analyst can get is the mean number of pilots and its confidence interval. With the arbitrary values that the authors used, the mean number of pilots was 38.69 with a standard deviation of 4.64. The upper 95 percent confidence interval was 39.26 and the lower 95 percent confidence interval was 38.12. The following figure shows the basic statistics for the mean number of pilots required for the operations (Figure 28).

Distributions

 # All Pilots



Quantiles

100.0%	maximum	50.330
99.5%		49.756
97.5%		46.654
90.0%		44.210
75.0%	quartile	41.770
50.0%	median	38.920
25.0%	quartile	36.400
10.0%		32.472
2.5%		28.214
0.5%		23.218
0.0%	minimum	22.420

Moments

Mean	38.691712
Std Dev	4.63863
Std Err Mean	0.2893498
upper 95% Mean	39.261521
lower 95% Mean	38.121903
N	257

Figure 28. Basic Statistics for the Mean Number of Pilots Needed for the Operations.

Another important statistic that simulation can provide is the composition of the pilot force. By composition, we mean the ratios of flight positions, LANTIRN categories, and weather categories.

We designate pilots' flight position as wingman, 2-ship lead, or 4-ship lead. Pilots start as a wingman and upgrade to other positions as their proficiency level increases. Figure 29 shows the percentages of the pilots:

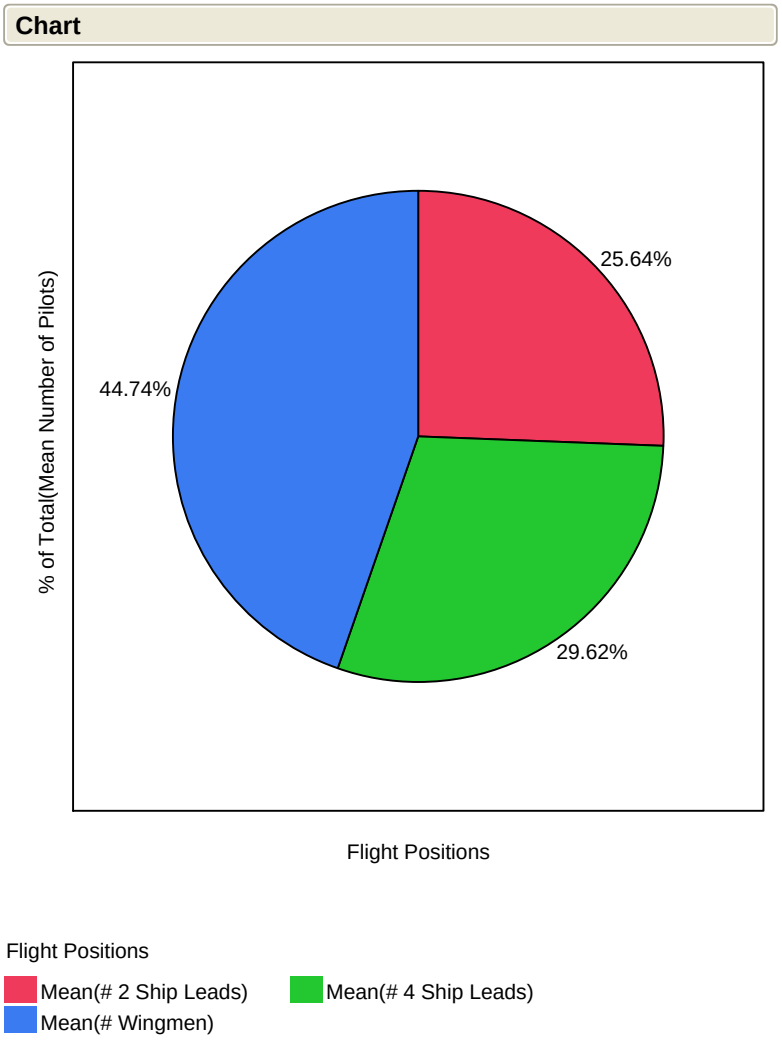


Figure 29. Percentages of Pilots Based on Flight Positions.

The analyst can also check the distribution of flight positions. This will give the analyst the means, standard deviations, and confidence intervals, as in Figure 30. Note that the scales differ.

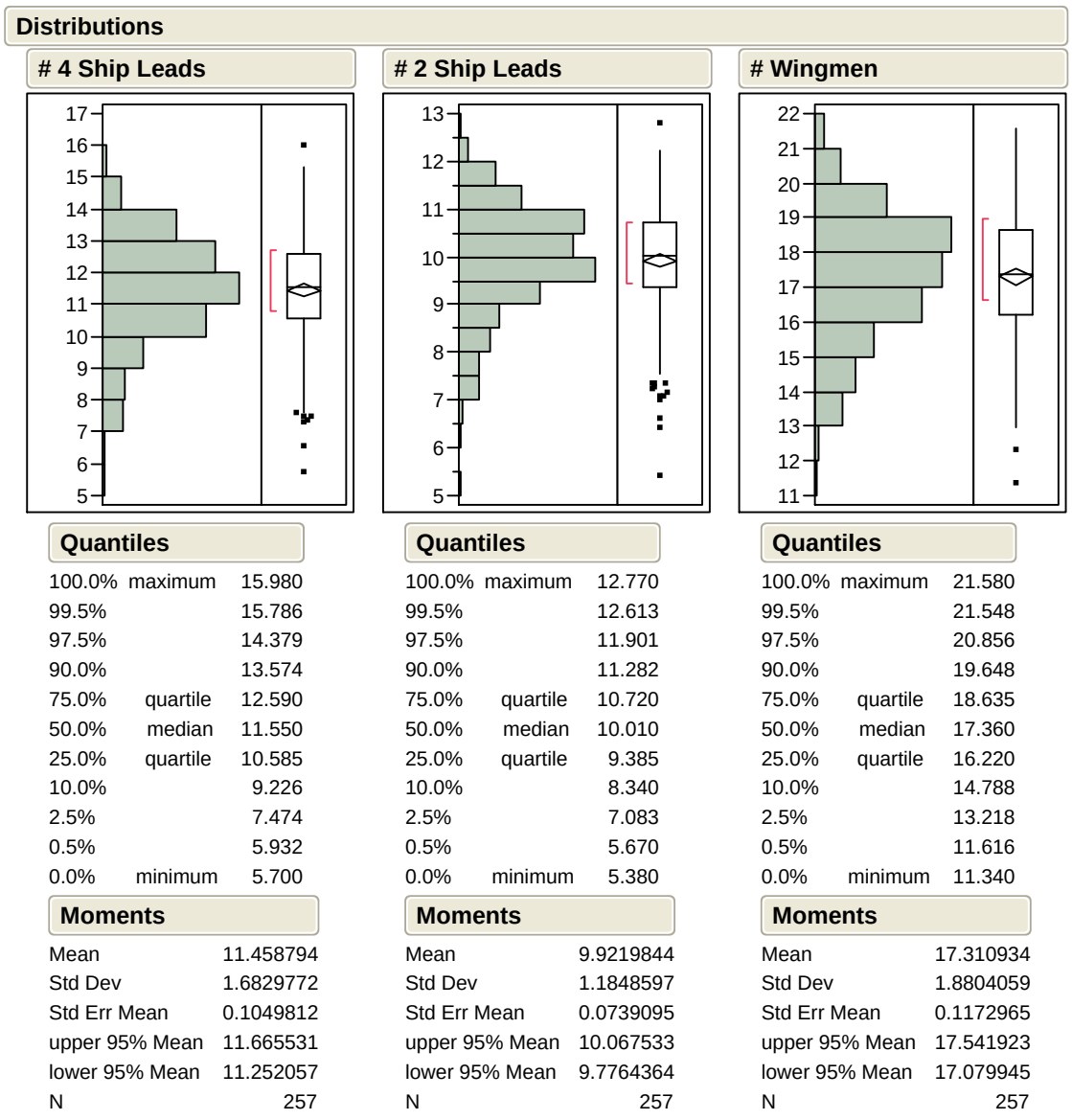


Figure 30. Distribution Output for Flight Positions.

The next statistic that analyst can check is LANTIRN category distributions within the squadron. This statistic can give an idea on the manpower planning as far as proficiency levels in LANTIRN.

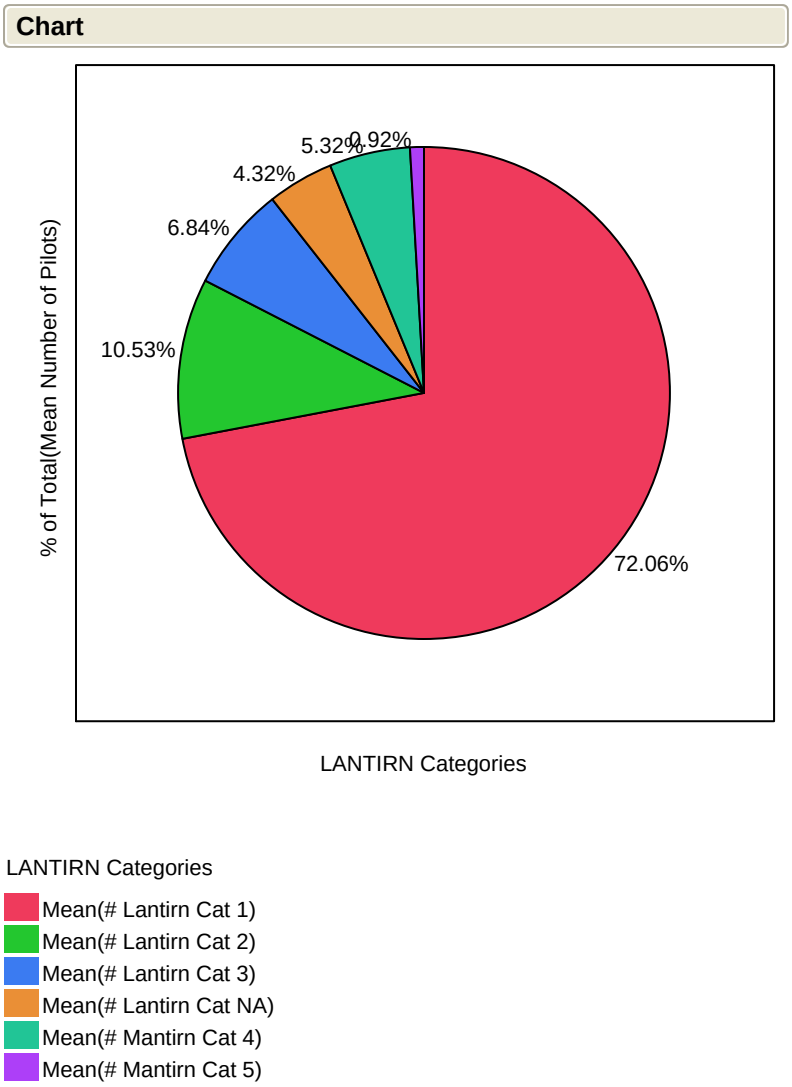


Figure 31. Percentages of LANTIRN Categories.

Like the flight positions, the analyst can check the distributions of LANTIRN categories. Figure 32 an example graph for this analysis. Note that the scales differ.

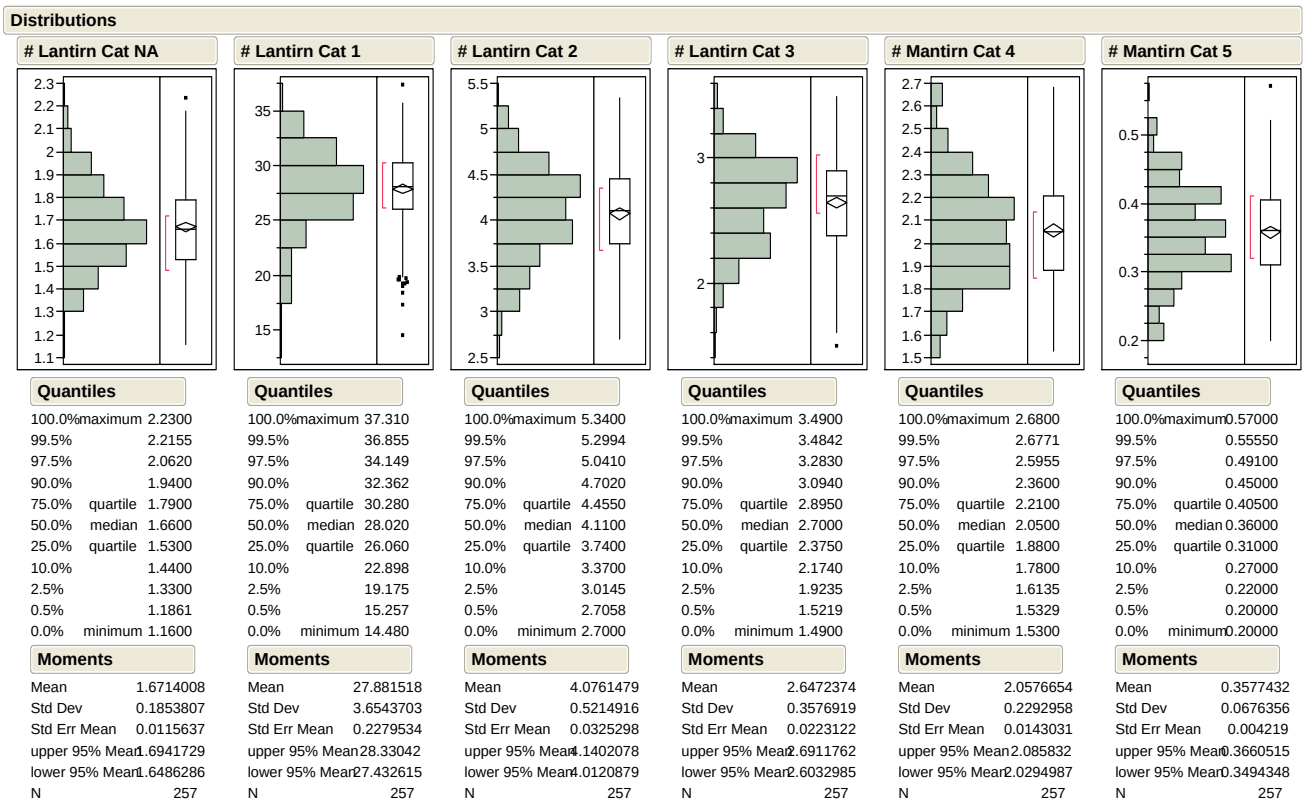


Figure 32. Distribution of LANTIRN and MANTIRN Categories.

Reader can see that majority of the pilots are LANTIRN category-1. Almost 90 percent of the pilots are LANTIRN Cat-1, Cat-2, and Cat-3. Nine percent of the pilots are MANTIRN Cat-4 and Cat-5, and less than 1 percent have no category.

The last composition of pilots' information that the analyst can look at is weather categories. There are three weather categories: Cat-3, Cat-2, and Cat-1. Cat-3 is the lowest level of proficiency. Weather categories generally accompany the flight positions. This is due to the reason that upgrade conditions are same for both of them.

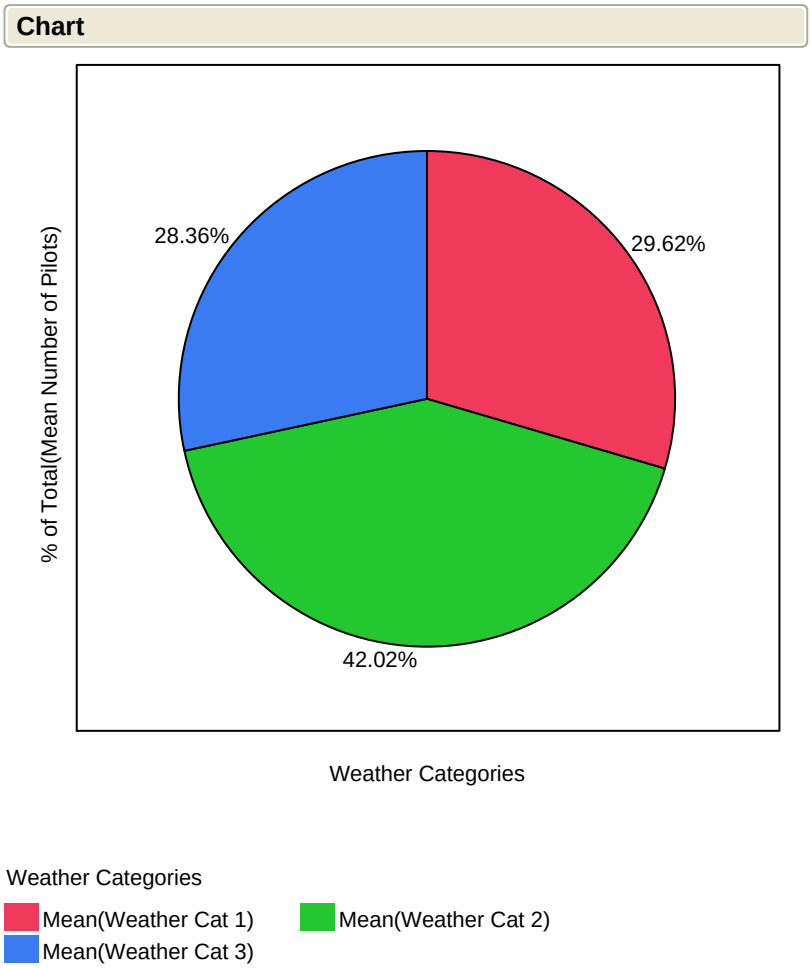
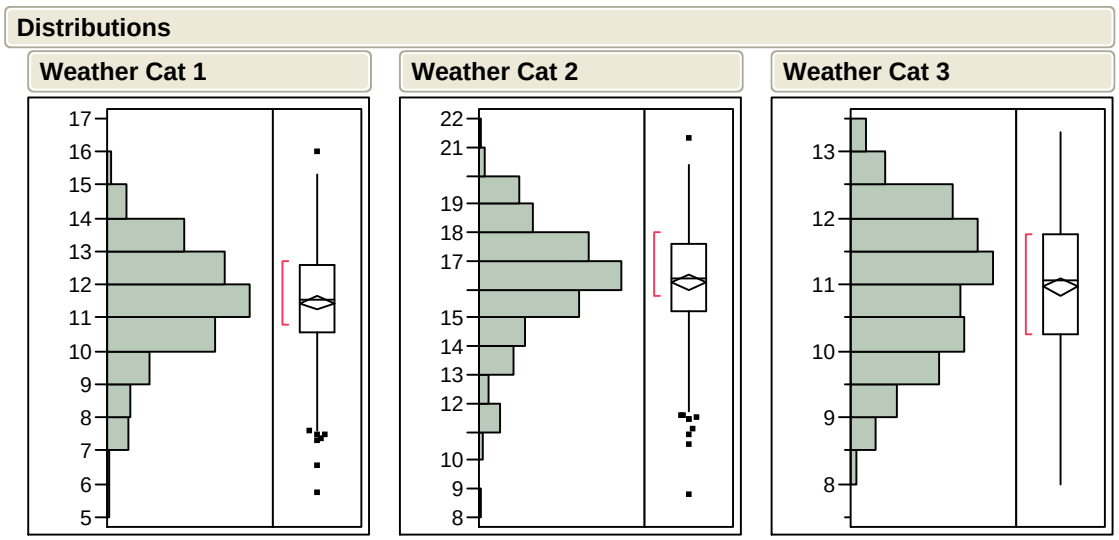


Figure 33. Percentages of Weather Categories for All Pilots.

The reader can note that majority of the pilots are weather category-2 pilots. Weather category-1 and weather category-2 pilots constitute approximately two-thirds of all pilots. Figure 34 shows the distribution data of weather categories.



Quantiles

		Weather Cat 1	Weather Cat 2	Weather Cat 3
100.0%	maximum	15.980	21.240	13.300
99.5%		15.786	20.991	13.262
97.5%		14.379	19.680	12.923
90.0%		13.574	18.726	12.374
75.0%	quartile	12.590	17.605	11.770
50.0%	median	11.550	16.370	11.060
25.0%	quartile	10.585	15.225	10.245
10.0%		9.226	13.360	9.500
2.5%		7.474	11.442	8.813
0.5%		5.932	9.236	8.049
0.0%	minimum	5.700	8.720	8.000

Moments

	Weather Cat 1	Weather Cat 2	Weather Cat 3
Mean	11.458794	16.259455	10.973463
Std Dev	1.6829772	2.0485451	1.0852715
Std Err Mean	0.1049812	0.1277847	0.0676974
upper 95% Mean	11.665531	16.511098	11.106778
lower 95% Mean	11.252057	16.007812	10.840148
N	257	257	257

Figure 34. Distribution Data for the Weather Categories of Pilots.

2. Multiple Regression Analysis

This simulation takes some parameters as inputs. After the simulation run is over, it gives a range of outputs. Input parameters are independent variables. They are also called explanatory variables. The output or the MOE (Measure of Effectiveness) is the dependent variable. It is also called the response variable. Regression analysis is a statistical tool that maps the relation between the

explanatory and the response variables. Simulation models are sometimes called “black boxes,” because most of the time we do not know the internal details of the model. The model takes some inputs and gives an output. Even though we do not know the internals of the model, we still can explore the relation between the inputs and the output by using regression analysis. This is an interactive process, and many different models are possible. Our intent is not to provide a detailed discussion of how to conduct regression analysis, but rather to provide an overview of some of the approaches that may be useful. Readers interested in more detail about regression model fitting should consult a statistics text like Deveaux, Velleman, and Bock (2005).

The following figure is the JMP regression analysis for a first-order (main-effects) regression model for the mean number of all pilots:

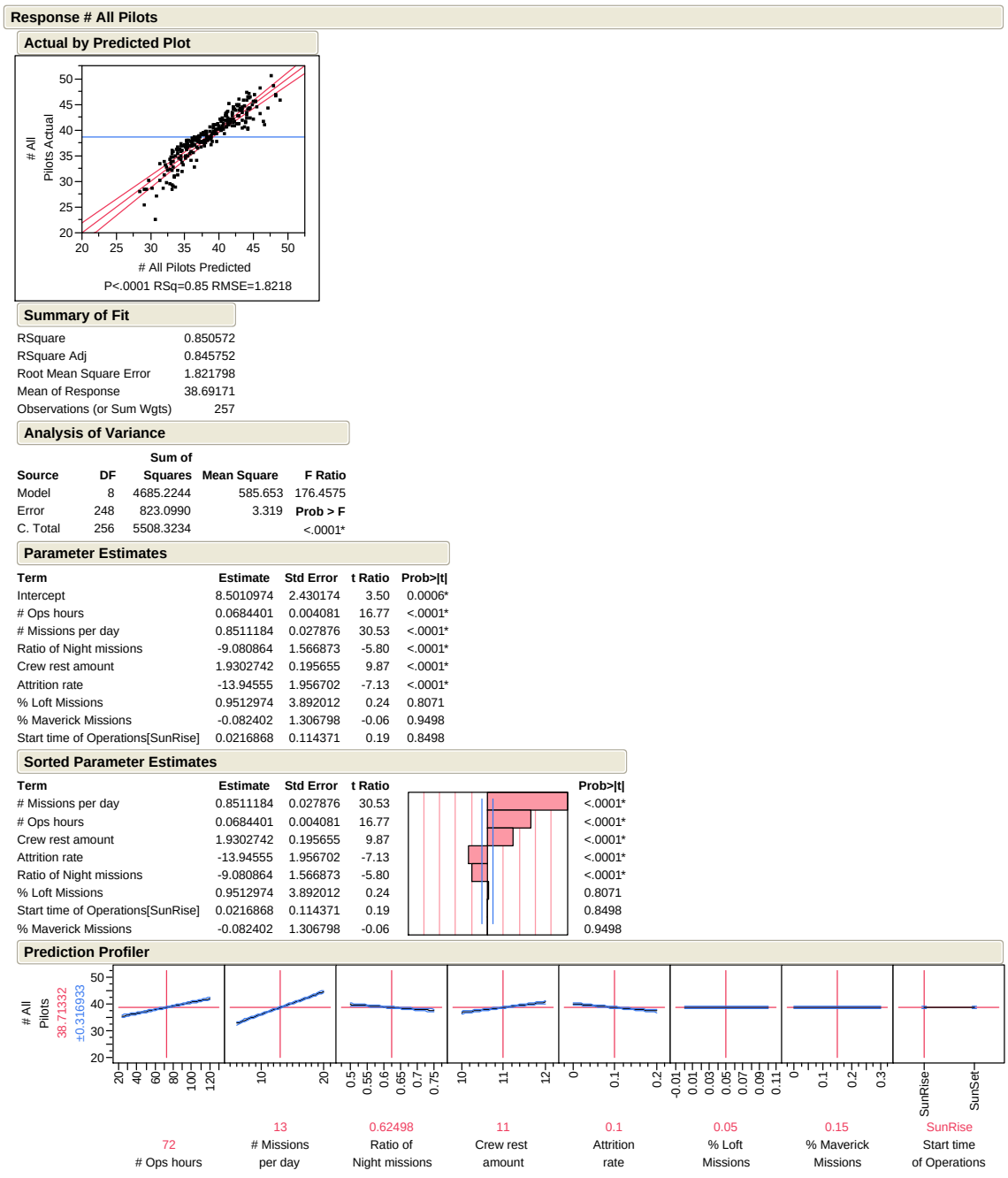


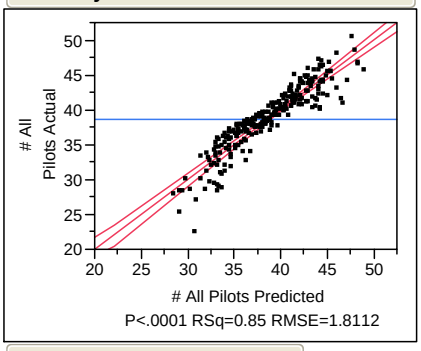
Figure 35. Main-Effects Regression Analysis Table for Mean Number of Pilots.

This model has only first-order terms. The reader will see that the percent loft missions, percent Maverick missions, and the start time of operations are not significant in the model. Therefore, it is safe to remove these terms from the

regression equation. Figure 36 shows that the RSquare and the RSquare Adjusted do not change in the model with these insignificant terms removed.

Response # All Pilots

Actual by Predicted Plot



Summary of Fit

RSquare	0.850515
RSquare Adj	0.847537
Root Mean Square Error	1.811221
Mean of Response	38.69171
Observations (or Sum Wgts)	257

Analysis of Variance

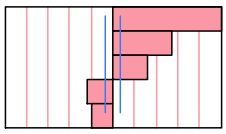
Source	DF	Sum of Squares	Mean Square	F Ratio
Model	5	4684.9127	936.983	285.6200
Error	251	823.4108	3.281	**Prob > F**
C. Total	256	5508.3234		<.0001*

Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Intercept	8.5233853	2.399072	3.55	0.0005*
# Ops hours	0.0684544	0.004057	16.88	<.0001*
# Missions per day	0.851198	0.027712	30.72	<.0001*
Ratio of Night missions	-9.099495	1.555572	-5.85	<.0001*
Crew rest amount	1.9325432	0.194262	9.95	<.0001*
Attrition rate	-13.96931	1.942707	-7.19	<.0001*

Sorted Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
# Missions per day	0.851198	0.027712	30.72	<.0001*
# Ops hours	0.0684544	0.004057	16.88	<.0001*
Crew rest amount	1.9325432	0.194262	9.95	<.0001*
Attrition rate	-13.96931	1.942707	-7.19	<.0001*
Ratio of Night missions	-9.099495	1.555572	-5.85	<.0001*



Prediction Profiler

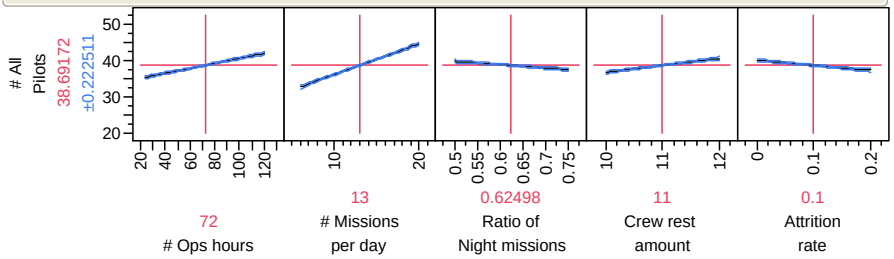


Figure 36. Main-Effects Multiple Regression Model With Insignificant Terms Removed.

R^2 is 0.85 and it means that 85 percent of the variability is explained by the predictor variables. This was a linear regression model. We can construct our regression model as follows:

Y = Mean number of pilots needed for the operations

$\beta_{\#}$ = constants/coefficients

X_1 = Number of operation hours

X_2 = Number of missions per 24 hours

X_3 = Ratio of night missions to all missions

X_4 = Crew rest amount

X_5 = Aircraft attrition rate

ε = error term

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \beta_4 X_4 + \beta_5 X_5 + \varepsilon$$

For the particular conditions that were used in this analysis, the multiple regression equation will be:

$$\hat{Y} = 8.523 + 0.068 X_1 + 0.851 X_2 - 9.099 X_3 + 1.932 X_4 - 13.969 X_5$$

The first plot to check is the residual by predicted plot. We should not see any patterns here. The residual by predicted plot in Figure 37 shows that the model is much more accurate for the middle of the predicted values than for the low or the high predictions. This suggests that even though the Rsquare is fairly high, a model that includes some interactions or quadratic effects may provide a better fit.

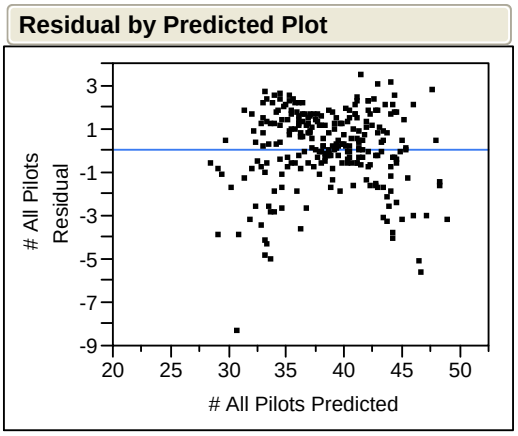


Figure 37. Residual by Predicted Plot.

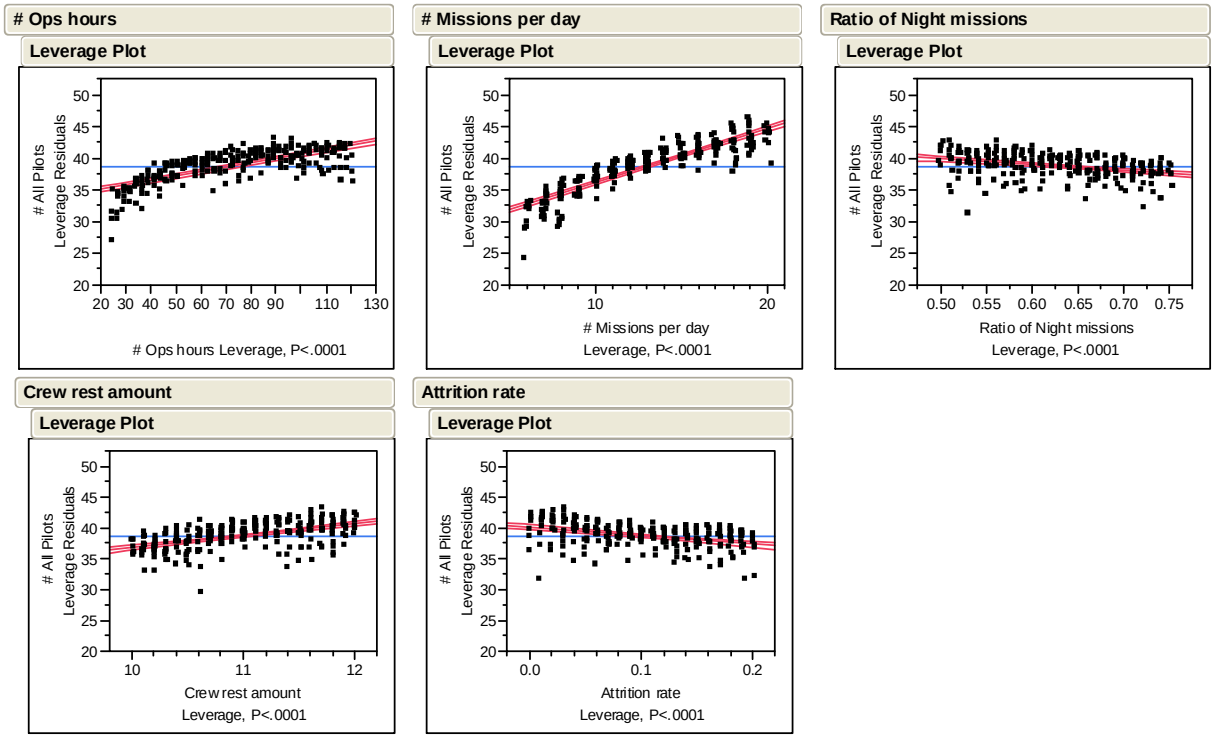


Figure 38. Leverage Plots.

Looking at leverage plots of predictor variables versus response MOE can provide important insights for the model. Figure 38 shows the leverage plots for the mean number of pilots. The first leverage plot is number of operations hours versus mean number of pilots. As the operation duration increases, the mean number of pilots also increases. This means that longer operation durations will

require a greater number of pilots. The second leverage plot is number of missions per day versus mean number of pilots. As the missions per day increase, the need for pilots also increases. The third leverage plot is the ratio of night missions to mean number of pilots. As the ratio of night missions to all missions increases, the mean number of pilots decreases. At first this might seem counter-intuitive, but it makes sense.

An increase in the ratio of night missions means more night missions and fewer day missions. This causes the night missions to arrive in quick succession. Pilots can fly another sortie before their up time is over. A lower ratio will cause the missions to spread to all night; hence, pilots will be only able to fly a single mission during their active time. The fourth leverage plot is the crew rest duration versus mean number of pilots. As the crew rest amount increases, the mean number of pilots also increases. That is a normal relation. More crew rest time means less active time for the pilots. This prevents pilots from executing multiple missions during their active time. In return, the simulation produces more pilots to fly the missions or to carry out the duties.

The last leverage plot is attrition rate versus mean number of pilots. As the aircraft attrition rate increases, the need for more pilots decreases. This is counter-intuitive. It is saying that if the squadron is fighting against an effective enemy, it will need fewer pilots. However, when the model is examined closely, it makes sense. At the beginning of the simulation, all aircraft are available. When missions arrive, the squadron is able to fly all the missions. This requires production of many pilots. Later in the simulation, when the aircrafts are lost, the squadron loses its ability to fly the arriving missions. Because many pilots are produced at the initial surge, the squadron will not need to produce that many pilots later in the simulation. On the other hand, if the attrition rate is very small, then the squadron will maintain its resources and will be able to fly arriving missions. If there are no pilots available at that time, new pilots will be produced. This will increase the mean number of pilots that the squadron needs.

3. Stepwise Regression Analysis

Another analysis that the future analyst can use is stepwise regression. In the following model, only one-way and two-way interactions are present. If a term does not add a significant amount of explanatory power to the model, this term is not added to the model. The significance threshold in this analysis is 0.05.

Stepwise Fit

Response: # All Pilots

Stepwise Regression Control

Prob to Enter 0.050
Prob to Leave 0.050
Direction: Mixed
Rules: Combine

Step History

Step	Parameter	Action	"Sig Prob"	Seq SS	RSquare	Cp	p
1	(# Ops hours-72)*(# Missions per day-13)	Entered	0.0000	4173.742	0.7577	658.33	4
2	(Crew rest amount-11)*(Attrition rate-0.1)	Entered	0.0000	513.5569	0.8509	315.18	7
3	(# Missions per day-13)*(Attrition rate-0.1)	Entered	0.0000	202.1763	0.8877	179.73	8
4	Ratio of Night missions	Entered	0.0000	112.2541	0.9080	105.41	9
5	(# Ops hours-72)*(Attrition rate-0.1)	Entered	0.0000	74.86566	0.9216	56.515	10
6	(# Ops hours-72)*(% Maverick Missions-0.15)	Entered	0.0131	14.99946	0.9243	50.317	12
7	(Ratio of Night missions-0.62498)*Start time of Operations{SunSet-SunRise}	Entered	0.0290	11.96302	0.9265	46.184	14
8	(Crew rest amount-11)*Start time of Operations{SunSet-SunRise}	Entered	0.0190	9.109001	0.9282	41.991	15
9	(# Ops hours-72)*Start time of Operations{SunSet-SunRise}	Entered	0.0202	8.779598	0.9298	38.022	16
10	(Ratio of Night missions-0.62498)*(Crew rest amount-11)	Entered	0.0367	6.986322	0.9310	35.273	17
11	(# Missions per day-13)*Start time of Operations{SunSet-SunRise}	Entered	0.0438	6.419907	0.9322	32.908	18
12	(% Loft Missions-0.05)*Start time of Operations{SunSet-SunRise}	Entered	0.0454	9.6186	0.9339	30.369	20
13	(Ratio of Night missions-0.62498)*(Crew rest amount-11)	Removed	0.0539	5.76206	0.9329	32.286	19
14	(% Loft Missions-0.05)*(% Maverick Missions-0.15)	Entered	0.0248	7.78631	0.9343	28.993	20

Figure 39. Stepwise Fit for the Mean Number of Pilots.

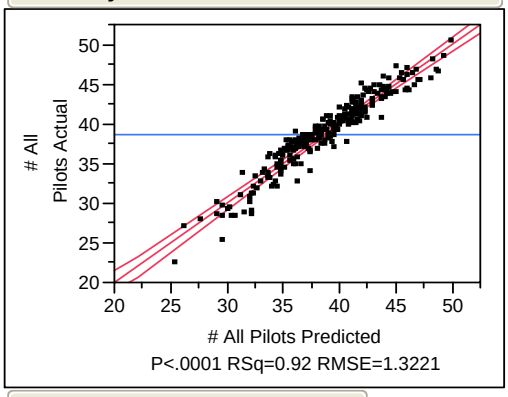
Because of the two-way interactions in the model, it has a higher R^2 value than the initial, main-effects model. There is a red line after the fifth term. This is due to fact that after the fifth term R^2 does not increase substantially with the addition of that extra term. For the sake of simplicity, only the first five terms can be used and still explain the 92.16 percent of the variability in the model. When compared to the previous multiple regression model, R^2 has gone up due to the use of two-way interactions. However, it is harder to explain the model. It is possible to construct a new multiple regression model with the addition of two-way interactions by hitting the “make model” button in JMP software. The model will be as follows:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + b_4 X_4 + \beta_5 X_5 + \beta_{12} X_1 X_2 + \beta_{45} X_4 X_5 + \beta_{25} X_2 X_5 + \beta_{15} X_1 X_5 + \varepsilon$$

The following graph shows the model constructed after stepwise model. The reader can see that there are nine terms included in the model. The R^2 value is 0.9216, as opposed to 0.85 for the model that has only main effects.

Response # All Pilots

Actual by Predicted Plot



Summary of Fit

RSquare	0.921623
RSquare Adj	0.918767
Root Mean Square Error	1.322077
Mean of Response	38.69171
Observations (or Sum Wgts)	257

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Ratio
Model	9	5076.5951	564.066	322.7129
Error	247	431.7284	1.748	**Prob > F**
C. Total	256	5508.3234		<.0001*

Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Intercept	8.5068426	1.751172	4.86	<.0001*
# Ops hours	0.0684544	0.002961	23.12	<.0001*
# Missions per day	0.851198	0.020228	42.08	<.0001*
Ratio of Night missions	-9.099556	1.13547	-8.01	<.0001*
Crew rest amount	1.9325432	0.141799	13.63	<.0001*
Attrition rate	-13.96931	1.418054	-9.85	<.0001*
(# Ops hours-72)*(# Missions per day-13)	-0.003708	0.000706	-5.25	<.0001*
(# Ops hours-72)*(Attrition rate-0.1)	-0.36973	0.056494	-6.54	<.0001*
(# Missions per day-13)*(Attrition rate-0.1)	-3.782744	0.359708	-10.52	<.0001*
(Crew rest amount-11)*(Attrition rate-0.1)	-7.331864	2.661035	-2.76	0.0063*

Sorted Parameter Estimates

Term	Estimate	Std Error	t Ratio		Prob>|t|
# Missions per day	0.851198	0.020228	42.08		<.0001*
# Ops hours	0.0684544	0.002961	23.12		<.0001*
Crew rest amount	1.9325432	0.141799	13.63		<.0001*
(# Missions per day-13)*(Attrition rate-0.1)	-3.782744	0.359708	-10.52		<.0001*
Attrition rate	-13.96931	1.418054	-9.85		<.0001*
Ratio of Night missions	-9.099556	1.13547	-8.01		<.0001*
(# Ops hours-72)*(Attrition rate-0.1)	-0.36973	0.056494	-6.54		<.0001*
(# Ops hours-72)*(# Missions per day-13)	-0.003708	0.000706	-5.25		<.0001*
(Crew rest amount-11)*(Attrition rate-0.1)	-7.331864	2.661035	-2.76		0.0063*

Figure 40. Multiple Regression Model with Significant Main Effects and Two-Way Interactions.

For the particular model here, the multiple regression equation will be like this:

$$\hat{Y} = 8.506 + 0.068X_1 + 0.851X_2 - 9.099X_3 + 1.932X_4 - 13.969X_5 - 0.0037(X_1 - 72)(X_2 - 13)$$
$$-0.369(X_1 - 72)(X_5 - 0.1) - 3.782(X_2 - 13)(X_5 - 0.1) - 7.331(X_4 - 11)(X_5 - 0.1)$$

4. Partition Tree

Another analysis tool that the analyst can utilize is the partition tree. The partition tree is a nonparametric tool “that recursively partitions the data to provide the most explanatory power for a performance measure of interest.” (Kleijnen, Sanchez, Lucas, & Cioppa, 2005)

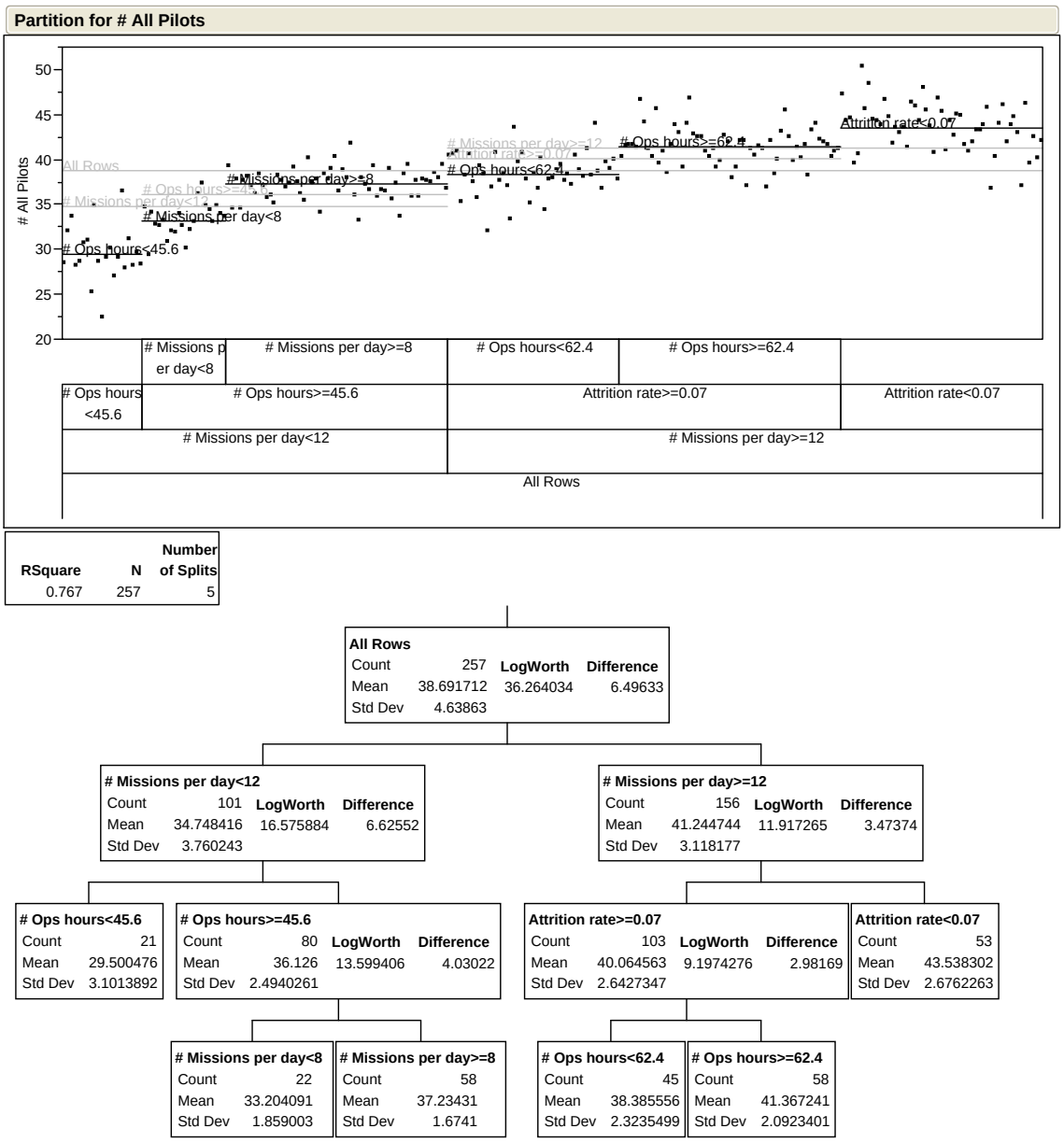


Figure 41. Partition Tree.

In this example, a partition is made until the R^2 is 0.767 (Figure 41). The particular area of interest is the region where the number of pilots produced is at minimum. This is to the left of the tree. When the missions per day is less than 12 and total duration of operations is less than 45.6 hours (approximately two days), the total number of pilots is at minimum (Mean 29.50 with a SD=3.1). It can be

concluded that if the squadron has less than 35 pilots, planners should strive to keep the length of the operation less than two days and keep the number of missions per day less than 12. On the right side of the partition tree, the highest mean number of pilots is 43.53 with a standard deviation of 2.67. It can be concluded that if the squadron is conducting operations against an ineffective enemy (i.e., the squadron does not lose aircraft that fast), and the number of missions per day is more than 12, then the squadron will need more than 40 pilots to sustain high operations tempo. As Kleijnen et al. stated, “Constructing a regression tree is an interactive process. Leaves are added until the analyst is satisfied that enough explanatory power is obtained, and splits can be forced at certain levels to examine smaller subsets of the data in more detail.” (Kleijnen et al., 2005) In this example, it was concluded that an R^2 value of 0.767 was enough. Making further splits was bringing little more explanatory power to the model, and it was partitioning the tree outside the area of interest.

THIS PAGE INTENTIONALLY LEFT BLANK

V. CONCLUSIONS AND RECOMMENDATIONS

We developed a simulation tool that can be used for defense manpower planning. The purpose of this tool is to use simulation techniques to find the necessary number and mix of pilot force for LANTIRN-equipped fighter squadrons. The greatest hindrance during the development of the model was lack of data. The data required to form the input probability distributions was inaccessible to the authors because of its classified nature. For this reason, we designed the simulation tool in a way that future users can supply correct and valid inputs to simulation via a graphical user interface. We also utilized a design of experiment concept. The research question stated that we needed to find the required number of pilots for various operations scenarios. The design of experiment allowed us to feed the simulation with varying levels of operations tempo. We also integrated effects of weather conditions into the model.

The GUI allows the analyst to select one of four design sizes for the experiment run. The simulation run is extremely fast. Even though a smaller size can be used, the authors recommend the use of largest design size with a high replication number. This will help tighten the confidence interval boundaries.

There is still ample room for further development of the model. We did not implement the aircrew sick calls in the model. Sick calls might reduce the actual number of pilots during operations. Another area that requires further research is loss of pilots at the airbase. We did not model the pilot attritions due to enemy air attacks to the base. Including this component will increase the realism level of the simulation. We used a non-homogenous Poisson process for the arrivals of sorties. However, arrival rates stay the same for the whole day or night. A better solution would be different arrival rates based on hours.

There is no optimization involved in the model. When a new mission arrives, simulation will search for a suitable pilot. The model tries to use existing pilots before it creates new ones. It can assign a 4-ship lead pilot to a wingman

position. A better model would take all the missions for the next day and accomplish assignment of existing pilots in the optimum manner before creating new pilots. This will require a change in model. The current model does not receive a list of the next day's missions. The new model will require a new implementation of receiving missions.

APPENDIX. METAR REPORT DESCRIPTORS

Qualifier		Weather Phenomena		
Intensity or Proximity 1	Descriptor 2	Precipitation 3	Obscuration 4	Other 5
- Light Moderate (no qualifier) + Heavy **VC** in the vicinity	**MI** Shallow **BC** Patches **DR** Low Drifting **BL** Blowing **SH** Showers **TS** Thunderstorms **FZ** Freezing **PR** Partial	**DZ** Drizzle **RA** Rain **SN** Snow **SG** Snow grains **IC** Ice Crystals (diamond dust) **PL** Ice Pellets **GR** Hail **GS** Small hail or snow pellets **UP** *Unknown Precipitation	**BR** Mist **FG** Fog **FU** Smoke **DU** Dust **SA** Sand **HZ** Haze **PY** Spray **VA** Volcanic ash	**PO** Dust/sand whirls **SQ** Squalls **FC** Funnel cloud **+FC** Tornado or Waterspout **SS** Sandstorm **DS** Dust storm

The weather groups are constructed by considering columns 1-5 in this table, in sequence; i.e., intensity, followed by descriptor, followed by weather phenomena; i.e., heavy rain showers(s) is coded as +SHRA.
* Automated stations only

Figure 42. Descriptors for the Weather Events in METAR Reports [From (FAA-H-8083-25, 2003)].

Sky Cover	Less than 1/8 (Clear)	1/8 - 2/8 (Few)	3/8 - 4/8 (Scattered)	5/8 - 7/8 (Broken)	8/8 or Overcast (Overcast)
Contraction	**SKC** **CLR** **FEW**	**FEW**	**SCT**	**BKN**	**OVC**

Figure 43. Sky Cover Contractions [From (FAA-H-8083-25, 2003)].

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

AFI 11-2f-16. (10 May 1996). *Air Force Instruction AFI 11-2f-16, F-16 Combat Aircraft Fundamentals, Volume 5.*

Bertsekas, D., & Tsitlikis, N. J. (2002). *Introduction to Probability* (Third Ed.). Massachusetts: Athena Scientific.

Brown, N., & Powers, S. (2000). Simulation In A Box (A Generic Reusable Maintenance Model). *Proceedings of the 2000 Winter Simulation Conference,* 1050-1056.

Buss, A. (1995). A Tutorial on Discrete-Event Modeling With Simulation Graphs. *Proceedings of the 1995 Winter Simulation Conference,* 74-81.

Buss, A. (2000). Component-Based Simulation Modeling. *Proceedings of the 2000 Winter Simulation Conference,* 964-971.

Buss, A. (2001). Discrete Event Programming with Simkit. *Simulation News Europe* (32/33), 15-25.

Buss, A. (Fall 2007). *MV4302 Advanced Discrete Event Simulation Modeling Class Notes.*

Buss, A. (Summer 2007). *OA3302 System Simulation Class Notes.*

Chatfield, C. (1991). *The Analysis of Time Series, an Introduction* (4th Ed.). London: Chapman and Hall.

Cioppa, M. T., & Lucas, W. T. (2007). Efficient Nearly Orthogonal and Space-Filling Latin Hypercubes. *Technometrics, 49*(1), 45-55.

De Veaux, D. R., Velleman, F. P., & Bock, E. D. (2005). *Stats: Data and Models* (1st Ed.). United States of America: Pearson Addison Wesley.

Faa-H-8083-25. (2003). *Pilot's Handbook of Aeronautical Knowledge.* U.S. Department Of Transportation, Federal Aviation Administration, Flight Standards Service.

German, M. K. (1990). Star-Eagle: An Expert Decision Support System for Assessing Tactical Air Capability. *Proceedings of the Fifth Annual AI Systems in Government Conference,* Washington DC. 134-139.

Harris, W. J. (2002). The Sortie Generation Rate Model. *Proceedings Of The 2002 Winter Simulation Conference,* 864-868.

Jane's Avionics. (2006). *Lantirn System.* Retrieved 11 April 2008 from http://www8.janes.com

Kleijnen, P. C. J., Sanchez, M. S., Lucas, W. T., & Cioppa, M. T. (2005). A User's Guide to the Brave New World of Designing Simulation Experiments. *Informs Journal On Computing, 17*(3), 263-289.

Law, M. A. (2007). *Simulation Modeling and Analysis* (4th Ed.). New York: McGraw-Hill.

Ozcan, D., email from Mr. Ozcan, Weather Office, Balikesir Air Base, Balikesir, Turkey to the author regarding METAR reports data for year 2007, 8 January 2008.

Sall, J., Creighton, L., & Lehman, A. (2005). *Jmp Start Statistics* (Third Ed.). Canada: Thomson.

Sanchez, M. S. (2006). Work Smarter Not Harder: Guidelines for Designing Simulation Experiments. *Proceedings of the 2006 Winter Simulation Conference,* 47-57.

SAS Institute Inc. (2007). *JMP 7.0.* Retrieved 01 January 2008 from www.jmp.com.

Zahn, A. E., & Renken, J. K. (1997). Eagle View: A Simulation Tool for Wing Operations. *Proceedings of the 1997 Winter Simulation Conference,* 917-924.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California

3. Hava Kuvvetleri Komutanligi
   BILKARDES Sube

   Ankara, TURKEY

4. Hava Harp Okulu Kutuphanesi
   Yesilyurt
   Istanbul, TURKEY

5. Kara Harp Okulu Kutuphanesi
   Ankara, TURKEY

6. Deniz Harp Okulu Kutuphanesi
   Tuzla
   Istanbul, TURKEY

7. Prof. Arnold H. Buss
   MOVES, Naval Postgraduate School
   Monterey, California

8. Prof. Susan Sanchez
   OR, Naval Postgraduate School
   Monterey, California

9. Prof. Enver Yucesan
   OR, Naval Postgraduate School
   Monterey, California

10. Mustafa Azimetli
    Merzifon
    Amasya, TURKEY